



Predicting Cryptocurrency Prices with Machine Learning Algorithms: A Comparative Analysis

Harsha Nanda Gudavalli
Khetan Venkata Ratnam Kancherla

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science. The thesis is equivalent to 10 weeks of full-time studies.

The authors declare that they are the sole authors of this thesis and that they have not used any sources other than those listed in the bibliography and identified as references. They further declare that they have not submitted this thesis at any other institution to obtain a degree.

Contact Information:

Author(s):

Harsha Nanda Gudavalli

E-mail: hagu22@student.bth.se

Khetan Venkata Ratnam Kancherla

E-mail: khka22@student.bth.se

University advisor:

Senior Lecturer Lawrence Henesey

Department of Computer Science(DIDA)

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

Abstract

Background. Due to its decentralized nature and opportunity for substantial gains, cryptocurrency has become a popular investment opportunity. However, the highly unpredictable and volatile nature of the cryptocurrency market poses a challenge for investors looking to predict price movements and make profitable investments. Time series analysis, which recognizes trends and patterns in previous price data to create forecasts about future price movements, is one of the prominent and effective techniques for price prediction. Integrating Machine learning (ML) techniques and technical indicators along with time series analysis, can enhance the prediction accuracy significantly.

Objectives. The objective of this thesis is to identify an effective ML algorithm for making long-term predictions of Bitcoin prices, by developing prediction models using the ML algorithms and making predictions using the technical indicators(Relative Strength Index (RSI), Exponential Moving Average (EMA), Simple Moving Average (SMA)) as input for these models.

Method. A Systematic Literature Review (SLR) has been employed to identify effective ML algorithms for making long-term predictions of cryptocurrency prices and conduct an experiment on these identified algorithms. The selected algorithms are trained and tested using the technical indicators RSI, EMA, SMA calculated using the historic price data over a period of May 2017 to May 2023 taken from CoinGecko API. The models are then evaluated using various metrics and the effect of the indicators on the performance of the prediction models is found using permutation feature importance and correlation analysis.

Results. After conducting SLR, the ML algorithms Random Forest (RF), Gradient Boosting (GB), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU) have been identified as effective algorithms to conduct our experiment on. Out of these algorithms, **LSTM** has been found to be the **most accurate model** out of the 4 selected algorithms based on **Root Mean Square Error (RMSE)** score (0.01083), **Mean Square Error (MSE)** score (0.00011), **Coefficient of Determination (R^2)** score (0.80618), **Time-Weighted Average (TWAP)** score (0.40507), and **Volume-Weighted Average (VWAP)** score (0.35660) respectively. Also, by performing permutation feature importance and correlation analysis it was found that the moving averages **EMA** and **SMA** had a greater impact on the performance of all the prediction models as compared to RSI.

Conclusion. Prediction models were built using the chosen ML algorithms identified through the literature review. Based on the dataset built from the data collected through the CoinGecko database and taking technical indicators as the input features, models were trained and tested using the chosen ML algorithms. The LSTM prediction algorithm was found to be the most accurate out of the chosen algorithms based on the RMSE, R^2 , TWAP, and VWAP scores obtained.

Keywords: Bitcoin, Cryptocurrency, Machine Learning, Price Predictions, Technical Indicators

Acknowledgments

We would like to convey our profound gratitude and appreciation to our supervisor, Lawrence Henesey, for his tremendous guidance, support, and encouragement over the course of our thesis project. We also want to express our gratitude to our family and friends for their constant support and inspiration over this academic year.

Contents

Abstract	i
Acknowledgments	iii
List of Acronyms	xiii
1 Introduction	1
1.1 Aim and Objective	2
1.1.1 Aim	2
1.1.2 Objectives	2
1.2 Research Questions	3
1.3 Ethical, societal and sustainability aspects	3
1.3.1 Ethical Aspects	3
1.3.2 Societal Aspects	3
1.4 Scope	3
1.5 Outline	3
2 Background	5
2.1 Machine Learning	5
2.1.1 Hyperparameters	5
2.1.2 Random Forest Algorithm	6
2.1.3 Gradient Boosting Algorithm	6
2.2 Deep Learning	7
2.2.1 Recurrent Neural Networks	7
2.2.2 Long Short-Term Memory	8
2.2.3 Gated Recurrent Unit	9
2.3 Technical Indicators	10
2.3.1 Relative Strength Index:	10
2.3.2 Simple Moving Average:	10
2.3.3 Exponential Moving Average:	11
2.4 Evaluation Metrics	12
2.4.1 RMSE:	12
2.4.2 MAE:	12
2.4.3 MSE:	12
2.4.4 R-Squared:	13
2.4.5 TWAP:	13
2.4.6 VWAP:	14

3	Related Work	15
4	Method	21
4.1	Literature Study	21
4.2	Experiment	22
4.2.1	Working Environment	22
4.2.2	Selecting the Dataset	23
4.2.3	Preprocessing the data	23
4.2.4	Creation of Prediction Models	24
4.2.5	Selection of Hyper-Parameters	25
4.2.6	Training The Models	26
4.2.7	Testing the Models	29
4.2.8	Feature Importance and Correlation Analysis	29
4.2.9	Evaluation Metrics	29
5	Results and Analysis	31
5.1	Systematic Literature Review Results	31
5.2	Experiment Result	37
5.2.1	Random Forest Regressor	37
5.2.2	Gradient Boosting Regressor	39
5.2.3	Sequential LSTM	42
5.2.4	Sequential GRU	44
5.3	Analysis	47
6	Discussion	49
6.1	Research Question 1	49
6.2	Research Question 2	50
7	Conclusions and Future Work	57
	References	59

List of Figures

5.1	Evaluation Results Bar Graph of Random Forest	37
5.2	Actual Vs Predicted Prices of Random Forest	38
5.3	Feature Importance Bar Graph of Random Forest	38
5.4	Correlation Analysis Bar Graph of Random Forest	39
5.5	Prediction Values of Random Forest Model	39
5.6	Evaluation Results Bar Graph of Gradient Boosting	40
5.7	Actual Vs Predicted Prices of Gradient Boosting	40
5.8	Feature Importance Bar Graph of Gradient Boosting	41
5.9	Correlation analysis Bar Graph of Gradient Boosting	41
5.10	Prediction Values of Gradient Boosting Model	42
5.11	Evaluation Results Bar Graph of LSTM	42
5.12	Actual Vs Predicted Prices of LSTM	43
5.13	Feature Importance Bar Graph of LSTM	43
5.14	Correlation Analysis Bar Graph of LSTM	44
5.15	Prediction Values of LSTM Model	44
5.16	Evaluation Results Bar Graph of GRU	45
5.17	Actual Vs Predicted Prices of GRU	45
5.18	Feature Importance Bar Graph of GRU	46
5.19	Correlation Analysis Bar Graph of GRU	46
5.20	Prediction Values of GRU Model	47
6.1	Actual Vs Predicted Prices of All models	53
6.2	Feature Importance Graph of All Models	54
6.3	Correlation Analysis Graph of All Models	54
6.4	Evaluation Results Graph of All Models	55

List of Tables

3.1	Related Work	20
5.1	SLR Findings	36
6.1	Evaluation Metric Values	51
6.2	Feature Importance Scores	52
6.3	Correlation Analysis Scores	52

List of Algorithms

1	Random Forest model training and evaluation	26
2	Gradient Boosting model training and evaluation	27
3	LSTM model training and evaluation	28
4	GRU model training and evaluation	28

List of Acronyms

AI	Artificial Intelligence
R^2	Coefficient of Determination
CNN	Convolutional Neural Networks
DL	Deep Learning
EMA	Exponential Moving Average
GRU	Gated Recurrent Unit
GB	Gradient Boosting
GDPR	General Data Protection Regulation
LSTM	Long Short-Term Memory
ML	Machine learning
MAE	Mean Absolute Error
MSE	Mean Square Error
NLP	Natural Language Processing
RF	Random Forest
RNN	Recurrent Neural Network
RSI	Relative Strength Index
RMSE	Root Mean Square Error
SMA	Simple Moving Average
SLR	Systematic Literature Review
TWAP	Time-Weighted Average
VWAP	Volume-Weighted Average

In recent years, cryptocurrencies have transformed the way people think about financial transactions and reshaped the traditional financial industry. The emergence of cryptocurrencies has given investors new opportunities to make successful investments [5]. Cryptocurrencies are virtual currencies that rely on blockchain technology, a distributed ledger system that enables safe and open exchanges without the need for a central authority [21]. The decentralized nature of cryptocurrencies makes them less vulnerable to manipulation by financial institutions than traditional currencies. Investors trying to precisely forecast price fluctuations and make profitable investments face problems due to the cryptocurrency market's extreme volatility and unpredictability [43]. To overcome this investors have employed different techniques like performing time-series, fundamental, sentimental, and technical analysis using ML algorithms.

One of the most popular and globally recognized cryptocurrencies is Bitcoin. Bitcoin was the first ever cryptocurrency to be created and was launched by a group of programmers under the pseudonym Satoshi Nakamoto in January 2009 [7]. The value of 1 Bitcoin has increased from \$357.24 in November 2015 to \$19,891.99 in December 2017, but again fell to a low of \$11,509.31 in October 2020 and then again rose to a high of \$30,220.42 in April 2023 [1]. This shows that investing in cryptocurrency offers the potential for significant gains but also carries the risk of substantial losses due to its volatile nature.

ML is a domain that deals with the development of algorithms and statistical models that enable computer systems to learn from data and improve their performance on a specific task. Some of the tasks of ML include classification, regression, anomaly detection, and Natural Language Processing (NLP) [35]. Making informed investment decisions can be facilitated by the application of ML in price prediction, which can offer useful insights into market dynamics [11]. With the use of historical data analysis, ML algorithms may spot patterns, trends, and correlations that human analysts would miss. ML models can capture trend and momentum information and forecast future price movements by utilizing a variety of techniques, including time series analysis [44] and incorporating technical indicators like the SMA, EMA, and RSI, among others [13].

The accuracy of price prediction in the cryptocurrency market can be improved by using technical indicators such as the SMA, EMA, and RSI as input [13]. Both

the SMA and EMA are trend-following indicators that are used to amplify price data and pinpoint the trend's direction. The momentum oscillator RSI, on the other hand, gauges how strongly prices fluctuate.

Many researchers have used ML algorithms for price prediction of cryptocurrency [34, 49]. As per our knowledge, there lacks a detailed comparative analysis of machine learning algorithms for long-term cryptocurrency price prediction where technical indicators like RSI, EMA, SMA are used as input features which is a significant gap in the current research. Utilizing technical indicators derived from historical data and forecasting future cryptocurrency prices, could assist investors in making informed investment decisions in this unpredictable market.

This thesis aims to explore the application of ML algorithms, including time series analysis, in predicting the future prices of cryptocurrencies, with a special focus on Bitcoin. In specific this thesis aims to explore and find out which ML algorithms are better suited for making long-term predictions of Bitcoin prices while using technical indicators as input features and conduct an experiment to identify the most accurate model among the identified ML algorithms. Bitcoin being the oldest cryptocurrency to ever exist has a large amount of historic price data which would help in training the ML models to gain better performance [7]. It also aspires to find out the impact of each input features on the ML models' performance.

1.1 Aim and Objective

1.1.1 Aim

The aim of this thesis is to compare the machine learning algorithms for the price prediction of Bitcoin while using technical indicators as inputs. The comparison of the algorithms will be done based on chosen evaluation metrics to find out which features in the dataset has the most significant effect on the predictions of each model.

1.1.2 Objectives

The objectives of this thesis are:

1. To find out the ML algorithms that can be used for the prediction of Bitcoin prices.
2. To compare the predictions of each ML algorithm using various evaluation metric scores.
3. To find out which among the technical indicators taken as input has more effect on the prediction of each model.

1.2 Research Questions

RQ1: Which ML algorithms are better suited to predict the prices of Bitcoin?

RQ2: Which among the selected ML algorithms performs the best in predicting the price of Bitcoin when using technical indicators RSI, EMA, SMA as input, and which among these indicators affects the prediction performance of each ML algorithm the most?

1.3 Ethical, societal and sustainability aspects

1.3.1 Ethical Aspects

To make certain of the ethical aspects, the data which was used for this comparative analysis does not contain any of the individual's personal data but rather consists of the daily prices of the cryptocurrency Bitcoin and the data chosen was extracted from an open-source website Coingecko that is accessible to everyone and complies with the General Data Protection Regulation (GDPR) policies [14].

1.3.2 Societal Aspects

To enhance the interpretability of our models' prediction results, feature analysis is conducted to determine which features in the dataset had the most significant impact on the predictions, this could assist investors in making more informed investment decisions.

1.4 Scope

This thesis aims to identify and compare ML algorithms that can predict the price of Bitcoin using technical indicators as input. It also strives to find out which among the technical indicators has more effect on the prediction of each model.

1.5 Outline

The structure of the thesis is as follows:

The chapter 2 discusses the technical terms which would help understand the thesis better, the chapter 3 discusses the work related to the field of study of this thesis, the chapter 4 elaborates the methodologies used to conduct this thesis, the chapter 5 shows the results after the execution of the thesis, the chapter 6 discusses the results and explains what those results mean, and finally, the chapter 7 concludes the thesis and presents the potential future work.

Concepts linked to the research for this thesis are covered in this section. It provides a quick introduction to the idea of machine learning and talks about the pertinent algorithms utilized. The evaluation metrics and technical indicators are also highlighted.

2.1 Machine Learning

ML, a sub-field of Artificial Intelligence (AI), is an interdisciplinary field that focuses on the development of computational algorithms and models that can autonomously learn and improve from data [35]. It encompasses a set of statistical techniques and algorithms that enable computer systems to analyze and interpret complex patterns and relationships within data, without being explicitly programmed.

ML has a wide range of applications, some of the included fields are: Computer vision- tasks like object detection, and object recognition. Prediction- tasks like the prediction of future trends, classification, analysis, and recommendation based on historic data [51]. NLP involves the analysis and understanding of human language. ML techniques are used for tasks such as sentiment analysis, text classification, language translation, chatbots, voice assistants, and text generation.

ML has made considerable strides in recent years thanks to the availability of big datasets, high computing power, and innovations in algorithmic techniques like Deep Learning (DL), AI, etc. [51].

2.1.1 Hyperparameters

The ML models have certain variables that determine the learning process, they are known as Hyperparameters. The selection of these variables has a significant effect on the model's performance. Therefore, the hyperparameters for the models must be selected carefully. Hyperparameter tuning is the process where the model is tested with varied values of hyperparameters to find out the most optimal parameters for the model [54].

2.1.2 Random Forest Algorithm

RF is a machine learning algorithm that is frequently employed for problems related to regression and classification [8]. The RF technique creates a forest of decision trees, with each tree trained on a different portion of the data set using a different subset of the parameters. This is done to increase volatility and lessen overfitting.

One of the fundamental characteristics of RF is the random selection of parameters at each split, which makes them efficient and noise-free. Initially, a random subset of the training set's data is chosen with replacement to build an RF. This process is referred to as Bagging. By adding randomness, bagging seeks to lower the variance of the model. Each tree is then trained using this bootstrap sample of the data. The approach evaluates a random subset of the features at each node of a decision tree to determine the optimal split [33].

By increasing the diversity and decreasing the association between the trees, the overall accuracy of the forest is increased. After each tree has been constructed, the algorithm predicts the class or regression value by combining the results of each tree. When performing classification tasks, the forest produces the mode of the classes predicted by each tree, whereas when performing regression tasks, the forest produces the mean of the predictions [33].

RF is scalable and can handle large datasets with millions of observations and thousands of features. One of the key features of RF is its ability to perform feature selection and feature importance analysis. This allows analysts to identify which input features are most relevant for predicting the price of a cryptocurrency. By focusing on the most important features, analysts can build more accurate and interpretable models, and avoid overfitting and model complexity [50]. To make a new prediction at new point x , we use the equation [19] :

$$\text{Regression: } \hat{f}_{rf}^J(x) = \frac{1}{J} \sum_{j=1}^J T_j(x)$$

Where,

$\hat{f}_{rf}^J(x)$ represents the predicted value of the target variable for a given input variable x .

J represents the number of decision trees in the forest,

$T_j(x)$ represents each decision tree in the forest.

2.1.3 Gradient Boosting Algorithm

GB is a robust ML method that is frequently employed across various industries to address a variety of challenges. It is a member of the ensemble approach family, which integrates several models to produce predictions that are more accurate than those produced by any one model alone [38]. GB involves adding models to an ensemble one at a time, with each new model aiming to fix any errors caused by

the previous one. To put it simply, it assembles a group of inefficient learners, like decision trees, and trains them to reduce the errors of the prior model.

The term "gradient" relates to the gradient descent optimization algorithm, which is utilized to minimize the loss function in GB. It aims to minimize the loss function, which evaluates the variation between the target variable's predicted and actual values [18]. This residual error is then used to train the following decision tree. This method is repeated until the error is reduced to a minimum or a specific number of iterations is reached. Since each iteration is checked to reduce errors, it even helps in reducing overfitting. It can be used for both classification and regression applications and can handle both numerical and categorical input. It can be used for both classification and regression applications and can handle both numerical and categorical input.

GB can effectively deal with missing data and outliers as one of its benefits. This characteristic is highly beneficial in Bitcoin markets, where data can be limited and unreliable. Since the cryptocurrency market is extremely volatile and prone to sudden changes, the algorithm's ability to handle non-linear relationships between variables is crucial for price prediction [37].

2.2 Deep Learning

DL is a subfield of machine learning. It mainly focuses on training neural networks with multiple levels to identify and represent complex patterns and relationships in the data [52]. Since deep learning deals with a range of neural network architectures, it is also known as Deep Neural Networks. DL models are designed to automatically extract hierarchical features from input data, enabling them to learn intricate patterns and relationships.

DL is effective for deciphering and obtaining knowledge from both massive amounts of data and data gathered from many sources [15]. Some popular DL architectures include Convolutional Neural Networks (CNN)s for image and video processing, Recurrent Neural Networks (RNNs) for sequential data processing, and Generative Adversarial Networks (GANs) for generating new data samples.

2.2.1 Recurrent Neural Networks

Recurrent Neural Network (RNN) is one of the most popular deep learning architectures and is used for a variety of tasks, including speech recognition, time series forecasting, creating image descriptions, video tagging, and many more [48]. RNN as the name suggests has cycles where the information is transmitted back into itself, thereby taking into account the previous input along with the current input. This enables RNNs to handle sequential data. When training an RNN, the network's parameters (weights and biases) are adjusted through a process called back-propagation, where the error signal is propagated backward from the output to the input layers. During this process, gradients are calculated, representing the rate

of change of the error with respect to the network's parameters. These gradients are then used to update the parameters and improve the network's predictions.

Even though RNNs have successfully overcome the limitations of feed-forward neural networks, they suffer from the vanishing gradient problem where the gradients calculated during back-propagation diminish or "vanish" as they propagate backward through the layers of the network [56].

2.2.2 Long Short-Term Memory

RNN architectures such as LSTM are frequently employed in time-series forecasting, speech recognition, and NLP because they can capture long-term dependencies. While traditional RNNs, experience the vanishing gradient problem. On the other hand, LSTM uses a memory cell to store and retrieve input from earlier time steps, which makes it suited for modeling sequential data with long-term dependencies [20].

LSTM consists of 4 main components: 3 component gates, namely the Input, Output, and Forget gates and a memory cell. The input gate regulates the entry of fresh inputs into the memory cell. Forget gate decides how much of the previous memory state should be forgotten by the memory cell. The output gate regulates the output of memory cell states, and finally, the memory cell is in charge of retaining the data and transmitting it to the next time step. Tanh and sigmoid activation functions are used to create the LSTM gates. The sigmoid function returns values between 0 and 1, to represent the opened or closed status of the gates. The tanh function returns values ranging from -1 to 1, to represent the strength of the memory cell state [31].

Backpropagation through time is used to minimize the loss function as the LSTM model is being trained to determine the best values for its parameters. The parameters include weight matrices and bias vectors of each gate along with the output layer. Gradient clipping can be used to limit the size of the gradient during training, as LSTMs can still suffer from vanishing gradients if the gradient is too small, and from exploding gradients if the gradient is too large. In order to predict future prices, LSTMs can be used to identify patterns in historical price data. The model uses a series of historical prices to forecast each subsequent price in the series. The loss function that calculates the difference between predicted and actual prices is optimized in order to train the LSTM [28].

The forward training process of an LSTM network can be described using the following equations [49]:

$$\begin{aligned}
 i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\
 f_t &= \sigma(W_f[h_{t-1}, x_t] + b_f) \\
 c_t &= f_t \cdot c_{t-1} + i_t \cdot \tanh(W_c[h_{t-1}, x_t] + b_c) \\
 o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\
 h_t &= o_t \cdot \tanh(c_t)
 \end{aligned}$$

Where x_t is the input at time step t , h_t is the hidden state at time step t , c_t is the cell state at time step t , and i_t , f_t , and o_t are the input gate, forget gate, and output gate, respectively, at time step t . W and b are the weight matrices and bias vectors, respectively. The sigmoid function and the hyperbolic tangent function \tanh are used to bind the output between 0 and 1, and between -1 and 1, respectively.

2.2.3 Gated Recurrent Unit

GRU is a form of RNN architecture that is frequently employed in NLP, speech recognition, and other sequential data analysis applications like price predictions. GRU is a type of RNN that processes sequential data by keeping an internal state that is changed whenever new input is added to the sequence [12]. GRU may, however, selectively recall or forget data from earlier time steps because it uses a gating mechanism to regulate the information's passage through the network. This gating mechanism also helps GRU to avoid the vanishing gradient problem.

GRU has two gating units: an update gate and a reset gate. The update gate chooses how much of the old state should be kept and how much new input should be added to the existing state. The reset gate regulates how much of the prior state is forgotten before absorbing the incoming input. These gates are developed through training and have the ability to modify their behavior based on the required task. GRU is simpler to implement and faster to train than LSTM because it has fewer variables and needs less computation. In comparison to other ML models, GRU has a number of benefits [55].

In order to estimate price trends in the extremely volatile and dynamic crypto market, it is first necessary to be able to capture long-term dependencies and patterns in the time series data. Additionally, because GRU can handle variable-length data sequences, it is appropriate for handling time series data with inconsistent time intervals or missing data points. GRU is able to analyze the enormous amount of data produced by the crypto market since it is extremely scalable and can be trained on massive datasets. To anticipate future price patterns, the model combines the extracted features with historical price data. The model's output is a list of predicted prices for a specific time [17].

The hidden state at time t , h_t , is updated based on the input at time t , x_t , and the previous hidden state, h_{t-1} , using the following equations [49]:

$$\begin{aligned} u_t &= \sigma(W_u[h_{t-1}, x_t]) \\ r_t &= \sigma(W_r[h_{t-1}, x_t]) \\ h_t &= (1 - u_t) \cdot h_{t-1} + u_t \cdot \tanh(W[r_t \cdot h_{t-1}, u_t]) \end{aligned}$$

Where u_t and r_t are update and reset gate, respectively

2.3 Technical Indicators

The technical indicators RSI, EMA, and SMA are generally used for financial analysis to understand market dynamics and generate trading signals [6, 22]. These indicators help identify the trend strength and potential price reversals. In this thesis, using these indicators as inputs to the ML models would help provide additional context and information beyond the raw price data. This would help the models identify complex patterns between the indicators and the prices enabling the models to achieve better performance.

These technical indicators are specifically utilized as input features in this thesis because they are well-established, frequently used, and have an established history of being effective in a wide range of market conditions. Additionally, they are simple to compute and analyze, so even a novice can understand them. These indicators can be used for both short-term and long-term studies, depending on the time range [36].

2.3.1 Relative Strength Index:

Technical analysis uses the RSI indicator to gauge the strength of price movement in financial assets [45]. Usually, it is determined by averaging the gains and losses on an item over a specified time period. Foretelling shifts in momentum and trend reversals are possible with the RSI. It is used by traders and analysts to identify probable overbought or oversold conditions in an asset as well as potential price corrections or reversals.

A ML model for the prediction can be trained using RSI as one of the input features. The model's capability to predict outcomes may be enhanced by adding RSI to its feature set [40].

The RSI is calculated using a mathematical formula that compares the magnitude of recent price gains to recent price losses over a specified period.

RSI calculation formula used in this thesis:

$$RSI = 100 - \left(\frac{100}{1 + RS} \right)$$

Where RS(Relative Strength) is Average Gain by Average Loss.

2.3.2 Simple Moving Average:

The average price of a security or asset over a given length of time is known as the "Simple Moving Average", and it is determined by summing up the prices for a given number of periods and dividing those values by the same number of periods [36]. SMA can be a helpful input element because it offers details on the price trend of the asset over a predetermined amount of time. This can assist in identifying both

short-term and long-term trends in the data, which can raise the predictive power of the machine learning model.

The SMA is calculated by adding together a specified number of prices over a given time period and then dividing the sum by the number of periods [6].

SMA calculation formula used in this thesis:

$$SMA = \frac{\sum_{i=1}^N P_i}{N}$$

Where N is the number of periods used in the calculation, P_i denotes the price of the asset at period i.

2.3.3 Exponential Moving Average:

EMA, which stands for Exponential Moving Average, is a moving average that, in contrast to a simple moving average, gives more weight to recent price data [36]. EMA is calculated by adding a weighted average of the current day's price and the previous day's price to the previous day's EMA value after applying a smoothing factor to it. The smoothing factor is a constant value that defines the weight given to the preceding EMA value.

EMA serves as a beneficial tool for long and short traders and investors because it responds more swiftly to price trends. As it aids in recognizing significant price swings and probable market turning points, it could be helpful for both long-term and short-term predictions [6].

The calculation of EMA involves two main components: the smoothing factor and the previous EMA value. The smoothing factor determines the weightage given to the most recent data point and is typically derived from a specified time period or the number of data points. Since it places more emphasis on recent data, the EMA reacts more quickly to price changes compared to the SMA [6].

EMA calculation formula used in this thesis:

$$EMA = (Price - EMA_{previous}) * SmoothingFactor + EMA_{previous}$$

Where $EMA_{previous}$ is the EMA value from the previous calculation "SmoothingFactor" is a constant that determines the weightage given to the most recent data point.

2.4 Evaluation Metrics

2.4.1 RMSE:

RMSE is a commonly used metric to evaluate the performance of a predictive model, particularly in regression analysis [24]. RMSE measures the difference between the predicted and actual values of a dataset and provides a measure of how much the predictions deviate from the actual values, on average. A lower RMSE value represents better performance.

The formula to calculate RMSE is:

$$RMSE = \sqrt{1/n * \sum((predicted_i - actual_i)^2)}$$

Where,

n is the number of data points in the dataset,

$predicted_i$ is the predicted value for the i -th data point,

$actual_i$ is the actual value for the i -th data point.

2.4.2 MAE:

Mean Absolute Error (MAE) is a commonly used metric for evaluating the accuracy of a predictive model. It measures the average magnitude of errors in a set of predictions, without considering their direction [24]. A lower MAE value suggests better performance.

The formula to calculate MAE is:

$$MAE = \frac{\sum_{i=1}^n |y_i - \hat{y}_i|}{n}$$

Where,

Y is the actual value,

\hat{Y} is the predicted value, and n is the number of observations.

2.4.3 MSE:

MSE is a commonly used metric to measure the difference between the predicted and actual values of a regression problem [24]. It is calculated by taking the average

of the squared differences between the predicted and actual values. A lower MSE value represents better performance.

The formula to calculate MSE is:

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

Where,

Y_i is the actual value,

\hat{Y}_i is the predicted value,

n is the number of observations.

2.4.4 R-Squared:

R^2 (pronounced as "R-squared") is a statistical measure that represents the proportion of the variance in a dependent variable that can be explained by the independent variables in a regression model [24]. R^2 is a measure of how well the regression model fits the data.

The value of R^2 ranges from 0 to 1, with higher values indicating a better fit of the model to the data. An R^2 value of 0 means that the model explains none of the variability of the response data around its mean, while an R^2 value of 1 indicates that the model perfectly predicts the response data.

The formula to calculate R^2 is:

$$R^2 = 1 - \frac{SSR}{SST}$$

Where,

SSR (Sum of Squared Residuals) is the sum of the squared differences between the predicted values and the actual values.

SST (Total Sum of Squares) is the sum of the squared differences between the actual values and the mean of the target variable.

2.4.5 TWAP:

TWAP is a metric used to measure the average price of an asset over a specific period of time [3]. It is commonly used in finance and trading to evaluate the performance of a trading strategy or to benchmark the execution of trades. The TWAP

value of the predicted prices closer to the TWAP value of the actual prices suggests that the models' performance is good.

The formula to calculate TWAP is:

$$\text{Weighted average price} = \frac{\sum_{i=1}^n \text{Price}_i \cdot \text{Time}_i}{\sum_{i=1}^n \text{Time}_i}$$

Where,

Price1 to Price n are the prices recorded for each time interval,

Time1 to Time n are the durations of the time intervals,

Total time is the sum of the durations of all the time intervals.

The formula used to calculate TWAP in this thesis is:

$$TWAP = \frac{\sum_{t=1}^T P_t}{\sum_{t=1}^T Q_t}$$

where,

P_t represents the price for a particular timestamp,

Q_t represents the actual test prices for the same timestamp,

and T represents the total number of timestamps.

2.4.6 VWAP:

VWAP is a financial indicator used to measure the average price at which a stock or other security has traded over a specific period, based on both the trading volume and price for each transaction. It is the sum of the total volume-weighted prices by the total traded volume for the given period [3]. The VWAP value of the predicted prices closer to the VWAP value of the actual prices suggests that the models' performance is good.

The formula to calculate VWAP is:

$$VWAP = \frac{\sum_{i=1}^n (\text{Price}_i \times \text{Volume}_i)}{\sum_{i=1}^n \text{Volume}_i}$$

Chapter 3

Related Work

For predicting the price of cryptocurrencies, there are numerous comparative studies available. Many researchers have conducted cryptocurrency price predictions using a variety of methods. Few of them employed DL models, while some just took ML techniques into account when conducting their research. Researchers have also taken into account various cryptocurrencies, like Ethereum and Litecoin. Few articles considered a short-term strategy, while others focused on long-term analysis. A few of these papers are discussed below:

Phaladisailoed et al. [42] used 1-minute interval trade data from January 1, 2012, to January 8, 2018, via the Kaggle website to predict Bitcoin values using the LSTM, GRU, Theil-Sen, and Huber regression models. Theil-Sen and Huber and other regression models were tested using the scikit-learn and Keras libraries, and the findings revealed that GRU and LSTM performed superior to them. With an R^2 of 0.00002 and an R^2 of 0.992, GRU had the best accuracy, however, it took more time than Huber regression. They have solely used R^2 metrics to assess their models. Evaluating using only linear regression models makes them unable to compute non-linear relations between variables. Comparably [4, 49] have also used RNN algorithms to conduct their studies.

In order to predict cryptocurrency prices, Seabe et al. [49] used three different types of RNN, including LSTM, GRU, and Bi-Directional LSTM. The study focuses on Bitcoin, ethereum, and Litecoin, three of the most popular cryptocurrencies, and measures the performance of the models using Mean Absolute Percentage Error (MAPE) and RMSE. According to the experimental findings, Bi-LSTM fared better than LSTM and GRU in terms of prediction accuracy, with the lowest MAPE values for Bitcoin, Litecoin, and Ethereum being 0.036, 0.041, and 0.124, respectively. However, they did not take into account any ensemble machine learning models and instead just compared deep learning algorithms. Only RMSE and MAPE measures were used to evaluate their models.

To determine which RNN model performed best, Vanderbilt et al. [4] tested 3 distinct cryptocurrencies—Bitcoin, Litecoin, and Ripple—using 3 different RNN models—Simple RNN, LSTM, and GRU. From January 2015 to March 2020, they used historical price information from coinmarketcap.com. In an effort to improve the model's accuracy, they have also run further tests using new data from Google Trends for the same time period. They came to the conclusion that the three pro-

cedures were comparable in accuracy to one another. Additionally, the accuracy did not improve further after conducting additional experiments utilizing the data from Google Trends. They simply used the RMSE statistic to evaluate their model, and they subsequently tried to improve the output by training them with each cryptocurrency's daily closing price including or excluding Google Trends information. To improve the validity of the forecast results, other evaluation metrics along with RMSE were used in this thesis.

Like [49] Kim et al. [29] have tested on three cryptocurrencies namely Bitcoin, Ethereum, and Litecoin using LSTM and GRU deep learning algorithms. The datasets were chosen from coinmarket based on kurtosis and skewness. LSTM and GRU models are trained and tested on the same hyperparameter configuration while increasing the number of epochs from 1 to 30. The accuracy of each model is measured by RMSE and MAE. After the testing phase, they concluded that GRU was more advantageous for the downward stabilization trend, and the LSTM was suitable for the upward stabilization trend. Similarly, other researchers [27,30] have used RMSE and MAE as their main evaluation metrics.

Iqbal et al. [27] focused on finding the most efficient technique out of ARIMA, FB Prophet, and XGBoosting for predicting the future price of Bitcoin based on RMSE, MAE, and R^2 parameters. After their experimentation on all three algorithmic techniques, they found out that the parametric score of ARIMA is the best of all three considered techniques. They have used the Kaggle dataset's historical price data and pre-processed that data and used it to build their models for comparison. They have mostly focused on supervised machine learning algorithms and have not considered any other type of algorithms for their comparison. On the other hand, our comparative analysis is based on a mixture of both Supervised learning models RF, GB and RNN models LSTM, GRU.

In order to determine the future trends of the Ethereum cryptocurrency, Kumar et al. [30] used deep learning techniques including the Multi-Layer Perceptron (MLP) and LSTM, and evaluated their findings using the RMSE, MAE, and metrics. They have made a long-term prediction and computed values for each day, hour, and minute using previous pricing data. They used daily data that included open price, closing price, high and low, volume per day, and hourly data from August 2015 to August 2018. They came to the conclusion that the LSTM model is superior to the MLP in terms of robustness and accuracy for long-term reliance. Our models were trained using the dataset that contained the technical indicators EMA, SMA, RSI based on the historical price data.

Like [27] few researchers have solely based their study on ML algorithms. Derbentsev et al. [16] focused on the short-term prediction model for price prediction of cryptocurrencies. The updated Binary Auto Regressive Tree (BART) was adapted to series data and standard models. Study shows that BART is more accurate than ARIMA. ARIMA model in slow-growing and transitional dynamic times. RMSE for this algorithm for the horizon of 14, 21, and 30 days was within the ranges 4%, 6%, and 8% respectively. They have taken closing prices from January 2017 to March

2019 according to yahoo finance, and calculated their time series in log return. They have made the log return for the next time period their target variable. The forecast was carried out on five different time horizons: 5, 10, 14, 21, and 30 days using three models for each cryptocurrency. In contrast, we have considered the values SMA, EMA, and RSI values as our target variables.

Chen et al. [9] have identified Bitcoin prices through regular prices and high-frequency prices to predict Bitcoin prices through ML techniques at different frequencies. In comparison with the usual price benchmark results, XGB and SDA machine learning algorithms have higher results with the highest statistical accuracy of 66% and 65.3% respectively. They have employed 2 datasets: The first includes the aggregated Bitcoin daily price, with a big interval and small scale, from CoinMarketCap.com. The second dataset consists of 5-minute interval Bitcoin real-time trading price data at high-frequency and large scale pulled from Binance. They have considered 12 features like block size, hash rate, etc, and used data from 2 different datasets to conduct their study whereas we have used technical indicators like (SMA, EMA, and RSI) and a single dataset from Coinbase to conduct our experiment on.

Similar to [9] Saad et al. [46] have used internal indicators mempool size, hash rate, etc. to study the trends of two cryptocurrencies namely Bitcoin and Ethereum, and found out the key network indicators affecting their price and then applied ML techniques like LSTM and Regression to find out the accuracy of their model. They have considered Bitcoin data from the “Blockchain” API and collected data from April 2016 to May 2018. Similarly, data related to Ethereum was collected using the information provided by an Ethereum exchange “Etherscan”, for the same time period. Their results showed that with LSTM, Bitcoin achieves higher accuracy with minimum error on each epoch. However, with the conjugate gradient method, the overall margin of error with the Hessian algorithm was more than the conjugate gradients. We have trained our models on the dataset containing the technical indicators (EMA, SMA, RSI) based on the historical price data.

Huang et al. [25] have used decision trees to predict Bitcoin prices using daily price data of BTC-USD from January 1st, 2012 to December 29th, 2017 at investing.com. They have split their data into 3 subsample sets for the purpose of calculating the technical indicators as inputs, training the decision trees, and testing the trained model. They have used 124 technical indicators to predict the range of the next-day returns. The 124 indicators are split into 21 non-overlapping return ranges. These technical indicators are included in the ta-lib library and are grouped into 5 categories by the ta-lib. They found that the proposed model has strong (out-of-sample) predictive power for narrow ranges of Bitcoin daily returns. We have only considered three technical indicators and predicted the future prices of Bitcoin.

Chowdhury et al. [11] have used ensemble learning methods, K-NN model, gradient boosted trees, and neural net model to find and analyze the close price of nine cryptocurrencies and for the index, cci30. They have collected historical data which includes seven-day week daily data for the month of January 2019 from coinmarketcap.com. to have considered 7 attributes and divided their dataset into two training

and testing. To find out the index values they have taken data from cci30.com. For forecasting the daily close price of the month of January 2019, they have used the K-NN algorithm in the RapidMiner platform. They obtained 92.4% accuracy using the ensemble learning method, and 90% accuracy from gradient-boosted trees and concluded that the ensemble learning method is considered as the best among all the models used in the paper. They have only used data for the month of January 2019 and found the corresponding index values using the K-NN model, whereas we used RF, GB, and GRU and predicted the future prices using data from the past 6 years.

Liu et al. [32] have used OLS and XGB to predict returns for 3703 cryptocurrencies for the 2013 – 2021 period. Based on daily data they have built an equal-weighted portfolio that gives 2.4% daily return with a 0.27 Sharpe ratio. They have used data from many varying sources ranging from databases, google Trends to bank databases, and other commodity prices. They obtained a 4.8% R^2 value, they came to the following conclusions: 1-day lagged returns have great predictive power for crypto returns, three OECD indices are important for forecasting cryptocurrency returns and Google searches have higher predictive power for large cryptocurrencies than small ones. They have used OLS and XGradient Boosting models to conduct their research and also built an equal-weighted portfolio. We have only considered 4 algorithms to predict the future prices of Bitcoin.

From the reviewed related work it can be observed that there is much work done using different algorithms and different datasets for the prediction of cryptocurrency prices, yet there is still much to be researched in terms of long-term predictions. Additionally, not many studies have used technical indicators for predicting cryptocurrency prices. By utilizing technical indicators this thesis aims to identify an efficient algorithm for making long-term predictions of Bitcoin prices.

The related work results are noted in Table 3.1

Reference	Technique	Outcome	Evaluation	Dataset
[27]	ARIMA, FBProphet, XGB	Bitcoin price prediction	RMSE: ARIMA: 322.4 FBProphet:229.5 XGB: 369	Kaggle

[32]	OLS, XGB	Forecasting returns for 3703 cryptocurrencies, creating equal-weighted portfolio	R^2 : XBG: 4.855% Returns: OLS: Equal-weighted: 6.297% Capital-weighted: 1.852% XGB: Equal-weighted: 7.105% Capital-weighted: 2.165%	Various sources
[11]	Ensemble learning methods, K-NN model, gradient boosted trees, neural net model.	Close price of cryptocurrency and their corresponding index	Accuracy: Ensemble learning: 92.4% Gradient boosted: 90%	Coin-marketcap, Cci30.com
[25]	Decision trees	Bitcoin price prediction	Average: Geometric: 2.77 Arithmetic: 4.20	Investing
[46]	LSTM, Regression trees	Cryptocurrency price prediction: Bitcoin, Ethereum	Accuracy: Bitcoin: 0.9957 Ethereum: 0.9999	Blockchain, Etherscan
[9]	LSTM, QDA, SVM, RF, XGB, LR, LDA.	Bitcoin price prediction	Accuracy: LR: 66.0% LDA: 63.9% QDA: 55.1% SVM: 65.3% RF: 51.0% XGB: 48.3% LSTM: 57.0%	Coin-marketcap Binance
[16]	ARIMA, Regression tree, BART	Cryptocurrency forecasting: Bitcoin, Ethereum, Ripple	RMSE of BART: 14 days: 4% 21 days: 6% 30 days: 8%	Yahoo finance
[30]	MLP, LSTM	Ethereum price prediction	RMSE: MLP: 21.3 LSTM: 20.53	CoinDesk

[29]	LSTM, GRU	Cryptocurrency price prediction: Bitcoin, Ethereum, Litecoin	RMSE: Type a: LSTM: 0.415 GRU: 0.091 Type b: LSTM: 0.268 GRU: 0.447	Coinmarket
[4]	Simple RNN, LSTM, GRU	Cryptocurrency price prediction: Bitcoin, Ripple, Litecoin	P-value (BTC, ETH, LTC): RNN: 0.997, 0.908, 0.704 LSTM: 0.939, 0.508, 0.877 GRU: 0.659, 0.729, 0.642	Google trends, coin-marketcap
[49]	LSTM, GRU, Bi-directional LSTM	Cryptocurrency price forecasting: Bitcoin, Litecoin, Ethereum	RMSE (BTC, LTC, ETH): LSTM: 1031.3, 148.5, 9.66 Bi-LSTM: 1029.3, 83.9, 8.0 GRU: 1274.1, 98.3, 8.1	Yahoo finance
[42]	LSTM, GRU, Theil-Sen and Huber regression	Bitcoin price prediction	R^2 : Theil-Sen regression: 99.17% Huber regression: 99.18% LSTM: 99.2% GRU: 99.2%	Kaggle

Table 3.1: Related Work

To identify the factors influencing the price of Bitcoin and develop an accurate prediction model for future prices, a combination of qualitative and quantitative research methods has been used. By doing so, it was possible to achieve a comprehensive understanding of the factors affecting Bitcoin prices and develop an efficient model for predicting future prices.

Qualitative research methodology in the form of a SLR has been employed, which involves a systematic search for relevant literature followed by a critical evaluation of the quality and relevance of the identified studies. The outcome of the SLR provides valuable insights into the current body of knowledge in the domain of research and helps in identifying suitable ML algorithms for the prediction of Bitcoin prices using technical indicators calculated from the historical price data.

Quantitative research methodology involves conducting an experiment to gather empirical data related to the predictions made on the historic Bitcoin price data. This experimentation involves building a model using the selected algorithms, training, testing, and evaluating them using the relevant dataset and appropriate metrics. The experimentation process helps to validate the model's accuracy and reliability.

4.1 Literature Study

To address research question 1, a qualitative analysis in the form of a structured literature review is conducted by following the guidelines provided by C.Wohlin [53]. This method helps in gaining an understanding of different ML algorithms for the prediction of cryptocurrency prices and also helps in identifying the appropriate algorithms for the prediction of Bitcoin using historical price data. The steps taken in our research are as follows:

1. Identifying Keywords: The keywords that were analogous to our research goal were identified- Machine Learning, Prediction, Deep Learning, Cryptocurrency, Technical Indicators, and Price.
2. Formulating the Search String: Different search strings were formulated using the keywords identified in the previous step.

3. Finding the Literature: Using the search string research literature was found in the databases- Google Scholar, IEEEExplore, Science Direct, arXiv.
4. Inclusion and Exclusion Criteria: After collecting the research papers, articles, and conference papers from the aforementioned databases using the search string, the inclusion and exclusion criteria were formulated.

Inclusion Criteria:

- (a) Literature related to ML algorithms for prediction tasks.
- (b) Articles need to be published.
- (c) Every literature must be in the English language

Exclusion Criteria:

- (a) Literature that is not completed.
 - (b) Articles that are not written in the English language.
 - (c) Literature related to classification.
5. Application of the inclusion and exclusion criteria: The inclusion and exclusion criteria were employed on the collected literature to filter them and finally the literature is selected.
 6. Reviewing the selected literature: The observations from the gathered literature are compiled and reviewed for analysis.

4.2 Experiment

In order to answer RQ2, a quantitative analysis in the form of an experiment was conducted using the algorithms identified from the SLR conducted to answer RQ1. The experiment is done by building prediction models using the selected algorithms. Thereby identifying the best algorithm for the prediction of Bitcoin prices.

4.2.1 Working Environment

- OS: Microsoft Windows 11 64-bit
- Processor: Intel CORE i5-1155G7 @ 2.50GHz
- Graphics: Intel IRISXe Graphics
- RAM: 16.00 GB DDR4 SDRAM
- Name: Visual Studio Code - Coding editor
- Name: python version 3.9.12 – Open-source programming language.
- Name: requests Version: 2.29.0 - Library to make HTTP requests.

- Name: pandas Version: 2.0.1 - Powerful data structures for data analysis [39].
- Name: pandas-ta Version: 0.3.14b0- Extension for pandas used for Technical Analysis Indicators [39].
- Name: scikit-learn Version: 1.2.2- A library that provides many regression machine learning algorithms [41].
- Name: keras Version: 2.12.0 – A library that provides many deep learning neural network models [10].
- Name: numpy Version: 1.23.5- Library that gives support for large multidimensional arrays and matrices [23].
- Name: matplotlib Version: 3.7.1- Library that provide plotting features [26].

The structure of the experiment was as follows:

1. Selecting and pre-processing the data set to obtain quality data and splitting them into three.
2. Performing Hyperparameter-Tuning using the validation dataset.
3. Training the ML models using the training dataset.
4. Testing the trained models using the testing dataset.
5. Evaluating the model using the selected metrics.
6. Finding out which feature in the dataset affected the prediction of all the models.

4.2.2 Selecting the Dataset

The historical price data in USD of Bitcoin was taken from the CoinGecko database which is a well-established and reputable open-source database that offers comprehensive and reliable historical price data of several famous cryptocurrencies like Bitcoin, Ethereum, and many more [2]. CoinGecko focuses specifically on cryptocurrency market data, making it highly relevant. Also, by requesting historical prices using an API helps in receiving data up to the date when the request was made.

4.2.3 Preprocessing the data

Models in this research use Bitcoin prices in USD from the CoinGecko database. In particular, the daily interval price data in USD from May 2017- May 2023, is focused. The price of Bitcoin had a significant shift in 2017, compared to the smaller changes observed in the previous years. The chosen period includes historical price data spanning six years, which would provide the machine learning algorithms enough data to train and test enabling them to produce effective results. The data is stored as a data frame after being extracted as a .Json file utilizing the API endpoint. The dataset initially contains the features price, volume, and their corresponding timestamps.

Data formatting is performed by utilizing the `to_numeric()` function for converting the prices and volumes data and `to_datetime()` function to convert the time column in the data frame. These formatting functions are called from the imported pandas library. Using the prices data, 3 technical indicators SMA, EMA, and RSI are calculated. The technical indicators EMA and SMA are calculated over a 5-day period, whereas the indicator RSI is calculated over a period of 14 days. These indicators are calculated with the help of `pandas_ta` library in python.

To handle the missing values all the NaN (not a number) spaces/values are replaced with 0, using the `fillna()` function in python. In order to make certain that all features are in the same scale, minmax scaler is used from the scikit-learn library. The minmax scaler transforms the values of each feature, between 0-1. This scaling technique preserves the relative relationships between the values while bringing them to a standardized range [42]. The minmax scaler uses the following formula to perform scaling operation:

$$X_{sc} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

After preprocessing the dataset, it consists of 6 features that are free from any missing values and ready for splitting the dataset into two:

1. Time
2. Price
3. Volume
4. SMA
5. EMA
6. RSI

The pre-processed dataset (which consists of 2190 rows and 6 columns) is now split for training, validating, and testing the ML models. The dataset is split in such a way that the training set contains 70% of the data from the start date taken which consists of 1532 rows, the Validation set contains 10% of the data which is 219 rows and the testing set contains the remaining 20% of the total data which constitutes 439 rows. These training, validation, and testing datasets are used to build and evaluate models from the chosen algorithms.

4.2.4 Creation of Prediction Models

Prediction models are needed to be built using the selected algorithms to train and test. The RF model is created by calling the `RandomForestRegressor()` function from the scikit-learn library. Similarly, the GB model is created by calling the `GradientBoostingRegressor()` from the scikit-learn library. The LSTM model is created using the Keras library by calling the `Sequential()` function and adding the LSTM units. Similarly, the GRU model is created using the Keras library using the `Sequential()` function and adding the GRU units.

4.2.5 Selection of Hyper-Parameters

For ML Algorithms to have higher performance, the most optimal hyper-parameters are needed to be selected. These hyper-parameters help the ML models to fit well to unseen data and help improve the performance of the models [54]. Hyper-parameter tuning is done with the help of validation dataset.

Random Forest Regressor:

The RF Regressor has a hyper-parameter called `n_estimators` which specifies the number of decision trees that form the forest. The regressor is trained using the training dataset in multiple iterations, with the number of estimators incrementing from 50 to 300 in steps of 50 at each iteration. In each iteration, the model makes predictions using the validation dataset, and the predictions are evaluated using the score. The model with the best score among all the hyper-parameter configurations is then selected for training the model. This approach ensures that the RF is optimized for the given dataset and provides the best possible performance.

Gradient Boosting Regressor:

Similar to RF the GB regressor has hyper-parameter `n_estimators` which represents the number of decision trees in the model. In addition to `n_estimators` it also has the hyper-parameter learning rate. It determines the contribution of each tree to the final prediction. To optimize the hyper-parameter, the model is trained using different combinations of hyper-parameters: learning rate (0.001, 0.01, 0.1) and `n_estimators` (50,100,150,200). The predictions are made using the validation dataset. The predictions of each iteration are evaluated using scores. The hyper-parameters of the model with the best scores are chosen for training the GB Regressor.

LSTM:

The LSTM model is configured with three hyperparameters: `batch_size`, `epochs`, and `lstm_units`. The batch size determines the number of samples processed before updating the model's internal parameters during training, while the epochs represent the number of times the learning process takes place. Lastly, the `lstm_units` specifies the number of units in each layer of the model. To optimize the LSTM model's performance, different combinations of the hyperparameters are evaluated using the training dataset. For each combination, predictions are made using the validation dataset, and the resulting scores are computed. After evaluating all possible hyperparameter combinations, the model with the least score is selected, and its hyperparameters are used to train the final LSTM model.

GRU:

The GRU model has the hyper-parameters: `batch_size(16, 32)`, which defines the number of samples processed before updating the internal parameters dur-

ing training, epochs (50, 100), which sets the number of learning iterations, and `gru_units` (50, 100), which specifies the number of units in each layer of the model. To find the best hyper-parameter combination, the GRU model is trained using the training dataset with all possible hyper-parameter combinations. The validation dataset is used to make predictions at each iteration, and the predictions are evaluated using the score. The hyper-parameters of the model with the lowest score are then selected to train the final model.

4.2.6 Training The Models

The models are trained using the training dataset and setting the hyper-parameters to the values found during validation.

Random Forest Regressor:

The RF algorithm builds several decision trees during training, with each tree being trained on a different random subset of the training data. The feature that offers the best separation of the data is used to make a split at each node of each tree. This process is repeated until each leaf node includes a subset of the training data, and the predicted value is the average of the target values in that subset. The model is built by calling the ‘`RandomForestRegressor()`’ class from the scikit-learn library and setting the number of decision trees to be created to 100. These decision trees are separately trained using random subsets derived from the training dataset. The model is trained using RSI, EMA, and SMA as the input features and ‘Price’ as the target variable.

Algorithm 1 Random Forest model training and evaluation

- 1: **Input:** Training data (X_{train}, y_{train}) , testing data (X_{test}, y_{test}) , hyperparameters $(n_{estimators})$
 - 2: **Output:** Random Forest model performance metrics
 - 3: Initialize Random Forest model rf with $n_{estimators}$ estimators
 - 4: Train model on training data (X_{train}, y_{train})
 - 5: Evaluate model on testing data (X_{test}, y_{test})
 - 6: Predict prices on testing data using model rf
 - 7: Compute Random Forest model RMSE, MSE, MAE, and R2 performance metrics
 - 8: Compute Random Forest model TWAP and VWAP performance metrics
 - 9: **return** Random Forest model performance metrics
-

Gradient Boosting Regressor:

GB chooses the optimal feature to divide the data at each node of the tree during training in order to generate a decision tree. By adding decision trees to the ensemble, the model is trained iteratively, with each new tree fixing the mistakes produced by previous trees. By calling the ‘`GradientBoostingRegressor()`’ class from the scikit-learn library and setting the number of decision trees to be created to 100. These decision trees are separately trained using random subsets derived from the training dataset. The model is trained using RSI, EMA, and SMA as the input

features and ‘Price’ as the target variable, which involves fitting the decision trees to the data using gradient descent to minimize the loss function.

Algorithm 2 Gradient Boosting model training and evaluation

- 1: **Input:** Training data (X_{train}, y_{train}) , testing data (X_{test}, y_{test}) , hyperparameters $(n_{estimators})$
 - 2: **Output:** Gradient Boosting model performance metrics
 - 3: Initialize Gradient Boosting model gb
 - 4: Set number of estimators to $n_{estimators}$
 - 5: Train model with training data (X_{train}, y_{train})
 - 6: Evaluate model on testing data (X_{test}, y_{test})
 - 7: Predict prices on testing data using model gb
 - 8: Compute Gradient Boosting model RMSE, MSE, MAE, and R2 performance metrics
 - 9: Compute Gradient Boosting model TWAP and VWAP performance metrics
 - 10: **return** Gradient Boosting model performance metrics
-

LSTM:

The LSTM neural network is built using Keras, a well-known DL library. The model is initially set up as an empty sequential model, allowing for the sequential addition of layers. A dense output layer with one unit and two LSTM layers, each with 50 `lstm_units`, is then added to the model. The loss function utilized is MSE, and the model is optimized using the Adam optimizer. Three technical indicators—RSI, EMA, and SMA—that are employed to forecast the price of Bitcoin make up the input data. After that, the input data is reconfigured to have three time steps and one feature per time step (3,1).

The target variable or Bitcoin price is represented by the output data, which is a 1D tensor with the same length as the input data. The fit method uses input data, target data, epochs (50) which tell how many times the model iterates through the complete training dataset, and batch size (32) which refers to the number of samples utilized in each training iteration, as inputs to train the model. Based on the difference between its predictions and the actual target values, the model modifies its weights and biases during training.

Algorithm 3 LSTM model training and evaluation

- 1: **Input:** Training data (X_{train}, y_{train}) , testing data (X_{test}, y_{test}) , hyperparameters $(n_{units}, n_{epochs}, n_{batch})$
 - 2: **Output:** LSTM model performance metrics
 - 3: Initialize LSTM model $lstm$
 - 4: Add LSTM layer with n_{units} units, return sequences, and input shape $(3, 1)$
 - 5: Add LSTM layer with n_{units} units
 - 6: Add Dense layer with 1 unit
 - 7: Compile model with optimizer Adam and loss function mean squared error
 - 8: Train model with n_{epochs} epochs and batch size n_{batch} on training data (X_{train}, y_{train})
 - 9: Evaluate model on testing data (X_{test}, y_{test})
 - 10: Predict prices on testing data using model $lstm$
 - 11: Compute LSTM model RMSE, MSE, MAE, and R2 performance metrics
 - 12: Compute LSTM model TWAP and VWAP performance metrics
 - 13: **return** LSTM model performance metrics
-

GRU:

Similar to the LSTM algorithm, GRU is also built using the Keras library. The training process of GRU is similar to that of LSTM, an empty sequential model is first created, and then 2 GRU layers with 50 units each and a dense output layer with 1 unit are added. The model is then fitted using input and target data. The input data which contains RSI, EMA, and SMA values, is reshaped to have 3 time steps and 1 feature per time step, yielding the 3D tensor of shape (number of samples, 1) as the input data. A 1D tensor of price values makes up the target data. The training is carried out with a batch size of 32 over 50 epochs.

Algorithm 4 GRU model training and evaluation

- 1: **Input:** Training data (X_{train}, y_{train}) , testing data (X_{test}, y_{test}) , hyperparameters $(n_{units}, n_{epochs}, n_{batch})$
 - 2: **Output:** GRU model performance metrics
 - 3: Initialize GRU model gru
 - 4: Add EMA layer with n_{units} units, return sequences, and input shape $(3, 1)$
 - 5: Add GRU layer with n_{units} units
 - 6: Add Dense layer with 1 unit
 - 7: Compile model with optimizer Adam and loss function mean squared error
 - 8: Train model with n_{epochs} epochs and batch size n_{batch} on training data (X_{train}, y_{train})
 - 9: Evaluate model on testing data (X_{test}, y_{test})
 - 10: Predict prices on testing data using model gru
 - 11: Compute GRU model RMSE, MSE, MAE, and R2 performance metrics
 - 12: Compute GRU model TWAP and VWAP performance metrics
 - 13: **return** GRU model performance metrics
-

4.2.7 Testing the Models

The testing process involves evaluating the performance of all the trained ML models on the same testing dataset. This testing dataset was obtained by splitting the pre-processed dataset, ensuring that the models are assessed on unseen data. The trained RF and GB models are used to make predictions using the `predict()` function from the Scikit-learn library and the testing dataset was used as the input. Similarly, predictions are made using `predict()` function from the Keras library on the trained LSTM and GRU models, and the input features of the test data are reshaped to match the input shape expected by the LSTM and GRU models. The predicted values by each model are stored in the test data data-frame for analysis.

4.2.8 Feature Importance and Correlation Analysis

In order to find out the importance of each feature in the prediction models. The permutation feature importance method was employed. Here, firstly the MSE (baseline) is computed for each model, then each feature is iterated and multiple permutations of the values of that feature are performed. For each permutation, the score of the model is calculated and stored in a list. The importance of each feature is then calculated as the difference between the original score and the mean of the permuted scores of that feature. This represents the change in the model's performance when the values of the features are randomly shuffled.

Correlation analysis is performed to find out the linear relationship between each feature and the prediction of each model. To calculate the correlation coefficients the '`corr`' method of the panda's library is used. It computes the pairwise correlation between each technical indicator and the prediction of each model.

4.2.9 Evaluation Metrics

Evaluation metrics are essential for determining a model's performance and its relevance for solving an identified problem. The choice of the evaluation metrics is based on the relevance of the problem to be solved and the characteristics of the target variables. In the case of Bitcoin price prediction, evaluation metrics are chosen based on their ability for determining the accuracy of the model's predictions. For this thesis, the metrics calculated are MSE, RMSE, MAE, and R-squared. The error metrics (MSE, RMSE, MAE) compute the difference between the actual and predicted prices, whereas R-squared measures the proportion of variance in the predicted prices. These metrics are calculated using the sklearn library. Additionally, TWAP and VWAP are used to find out the time-weighted and volume-weighted averages of the predicted prices. These metrics are calculated by using the formulas mentioned in section 2.10

Bitcoin is highly volatile, and its price is affected by many factors like market sentiments, the current volume of the Bitcoin, investor confidence, economic factors, and historical prices of Bitcoin. In this thesis, historical price data is considered for the prediction, because it allows us to identify trends and patterns that have

occurred in the past and can potentially reoccur in the future. Moreover, historical data provides a large amount of information that can be used to train ML models to make long-term predictions. It is also important to note that predicting Bitcoin prices using historical data is not a foolproof method, as there is no guarantee that past trends will continue into the future.

A SLR has been conducted to identify various ML algorithms for predicting the prices of Bitcoin. RSI, EMA, and SMA are calculated using the historical price data of Bitcoin data, and a dataset was created. RF, GB, LSTM, and GRU were trained with the dataset which contains these technical indicators as the input features and prices as the output variable. These algorithms were evaluated using the metrics (RMSE, , MAE, R^2 , TWAP, VWAP).

5.1 Systematic Literature Review Results

To identify ML algorithms that can be used to predict Bitcoin prices, a SLR was performed. The relevant literature was reviewed and the findings are tabulated in table 5.1:

S.No	Title	Findings
1	Time-Series Prediction of Cryptocurrency Market using Machine Learning Techniques [27]	This paper focused on finding out the most efficient technique out of ARIMA, FB Prophet and XG Boosting for predicting the future price of Bitcoin based on RMSE, MAE and R^2 parameter. After their experimentation on all three algorithmic techniques, they have found out that the parametric score of ARIMA is the best of all three considered techniques.

2	Predicting the Trends of Price for Ethereum Using Deep Learning Techniques [30]	<p>This paper focused on using deep learning techniques like Multi-layer perceptron (MLP) and Long-Short term memory (LSTM) to find out the future trends of Ethereum cryptocurrency based on the data from CoinDesk and coin market repository and assessed against RMSE, MAE, and parameters. They concluded that the LSTM model is more robust and precise for long-term dependency as compared to MLP.</p>
3	Toward Characterizing Blockchain-Based Cryptocurrencies for Highly Accurate Predictions [46]	<p>This paper focused on studying the trends of two cryptocurrencies namely Bitcoin and Ethereum and found out the key network indicators affecting their price and then applied machine learning techniques like LSTM and Regression to find out the accuracy of their model. Their results showed that with LSTM, Bitcoin achieves higher accuracy with minimum error on each epoch. However, with the conjugate gradient method the overall margin of error with the Hessian algorithm was more than the conjugate gradients.</p>
4	An Applied Study of RNN Models for Predicting Cryptocurrency Prices [4]	<p>This paper focused on experimenting on 3 RNN models (Simple RNN, LSTM, and GRU) on 3 different cryptocurrencies (Bitcoin (BTC), Litecoin (LTC), and Ripple (XRP)) to find out which performed better. They have also used additional data from google trends to perform additional experiments to try and boost the accuracy of the model. They concluded that the 3 techniques considered were similar in terms of accuracy to each other. Also, after performing additional experiments using the data from google trends the accuracy did not increase further.</p>

5	A Cryptocurrency Prediction Model Using LSTM and GRU Algorithms [29]	This paper focused on testing on three cryptocurrencies namely Bitcoin, Ethereum, and Litecoin using LSTM and GRU deep learning algorithms. The datasets were chosen from coinmarket based on kurtosis and skewness. After the testing phase they have concluded that, GRU was more advantageous for the downward stabilization trend, and the LSTM was suitable for the upward stabilization trend.
6	Forecasting Cryptocurrency Prices Time-Series Using Machine Learning Approach [16]	This paper focused on the short-term prediction model for machine-learning cryptocurrencies. The updated Binary Auto Regressive Tree (BART) was adapted to series data and standard models. Study shows that BART is more accurate than the ARIMA, ARFIMA model in slow-growing and transitional dynamic times. In particular, RMSE for this algorithm for the horizon of 14, 21, and 30 days was within the ranges 4%, 6%, and 8% respectively.
7	Bitcoin Price Prediction Using Machine Learning: An Approach to Sample Dimension Engineering [9]	This paper focused on identifying Bitcoin prices through regular prices and high frequency prices to predict Bitcoin prices through machine learning techniques at different frequencies. In comparison with the usual price benchmark results, XGB and SDA has higher results with highest statistical accuracy of 66% and 65.3% respectively.

8	Forecasting Cryptocurrency Prices Using LSTM, GRU, and Bi-Directional LSTM: A Deep Learning Approach [49]	<p>This paper focused on using three types of Recurrent Neural Networks (RNNs), namely Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Bi-Directional LSTM (Bi-LSTM), for forecasting cryptocurrency prices. The study focuses on three major cryptocurrencies, Bitcoin (BTC), Ethereum (ETH), and Litecoin (LTC), and evaluates the performance of the models using Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE). The experimental results show that Bi-LSTM outperformed LSTM and GRU in terms of prediction accuracy, with the lowest MAPE values of 0.036, 0.041, and 0.124 for BTC, LTC, and ETH, respectively.</p>
9	Machine Learning Models Comparison for Bitcoin Price Prediction [42]	<p>This paper focused on using LSTM, GRU, Theil-Sen and Huber regression models to predict Bitcoin prices using 1-minute interval trading data. Different regression models were experimented with using scikit-learn and Keras libraries and the results showed that GRU and LSTM performed better than Theil-Sen and Huber regression models. GRU showed the best accuracy with an of 0.00002 and R^2 of 0.992, but it took more time than Huber regression.</p>

10	Predicting Bitcoin Returns Using High-Dimensional Technical Indicators [25]	This paper focused on using decision trees to predict Bitcoin prices using daily price data of BTC-USD. They have used 124 technical indicators to predict the range of the next day returns. The 124 indicators are split into 21 non-overlapping return ranges. These technical indicators are included in the ta-lib library and are grouped into 5 categories by the ta-lib. They found that the proposed model has strong (out-of-sample) predictive power for narrow ranges of Bitcoin daily returns.
11	An Approach to Predict and Forecast the Price of Constituents and Index of Cryptocurrency Using Machine Learning [11]	This paper focused on using ensemble learning methods, K-NN model, gradient boosted trees, and neural net model to find and analyze the close price of nine cryptocurrencies and for the index, cci30. They have obtained 92.4% accuracy using ensemble learning method, 90% accuracy from gradient boosted trees and concluded that ensemble learning method is considered as the best among all the models used in the paper.
12	Forecasting Cryptocurrency Returns with Machine Learning [32]	This paper focused on using OLS and XGB to predict returns for 3703 cryptocurrencies for the 2013 – 2021 period. Based on daily data they have built an equal-weighted portfolio that gives 2.4% daily return with a 0.27 Sharpe ratio. They have obtained a 4.8% R^2 value, they came to the following conclusions: 1-day lagged returns have great predictive power for crypto returns, three OECD indices are important for forecasting cryptocurrency returns and Google searches has higher predictive power for large cryptocurrencies than small ones.

13	Predicting Gold and Silver Price Direction Using Tree-Based Classifiers [47]	This paper focused on using several machine learning tree-based classifiers (bagging, stochastic gradient boosting, random forests) to predict the price direction of gold and silver exchange traded funds. Decision tree bagging, stochastic gradient boosting, and random forests predictions of gold and silver price direction are much more accurate than those obtained from logit models. For a 20-day forecast horizon, tree bagging, stochastic gradient boosting, and random forests produce accuracy rates of between 85% and 90% while logit models produce accuracy rates between 55% and 60%. Stochastic gradient boosting accuracy is a few percentage points less than that of random forests for forecast horizons over 10 days.
14	Cryptocurrency Price forecasting: A Comparative Study of Machine Learning Model in Short-Term Trading [34]	This paper focused on present a comparative performance of large-scale selected, Machine Learning algorithms for cryptocurrency forecasting. Specifically, this paper concentrates on forecasting time series data for a short-term trading period in ten cryptocurrencies (BTC, ETH, etc.) with ten selected machine learning algorithms (Random Forest, K-Nearest Neighbours, Neural Networks, Gradient Boosting, etc.) Their experimental results show that the Gradient Boosting with the mean square error criterion is superior in predicting most major cryptocurrencies by performing statistical analysis and data visualizations. Additionally, the Random Forest and Decision Tree model built by the Classification and Regression Tree algorithm also shows outstanding performance.

Table 5.1: SLR Findings

The SLR conducted on these papers has shown that RF, GB, LSTM, and GRU have been proven effective in predicting cryptocurrency prices. So, these ML algo-

rithms have been chosen to conduct the experiment.

5.2 Experiment Result

An experiment is conducted using the ML algorithms identified from the results of the SLR. The algorithms RF, GB, LSTM, and GRU are trained using the same dataset and then tested for the prediction of Bitcoin prices. The predictions are evaluated using RMSE, MSE, MAE, R^2 , TWAP, and VWAP. For evaluation of the models' predictions using TWAP and VWAP, the TWAP and VWAP values were calculated using the actual prices of the Bitcoin. The TWAP value calculated using actual Bitcoin prices was found to be 0.39953 and the VWAP value calculated using actual Bitcoin prices was found to be 0.34847. The results are visualized using the Matplotlib library for creating line graphs and Microsoft Word for creating bar graphs. The experiment results are presented in this section.

5.2.1 Random Forest Regressor

Hyper-parameter tuning was performed on RF with varying `n_estimators` values, it was found that the model performed the best when the `n_estimators` was set to 100 with the validation set score of 0.00052. Therefore, the `n_estimators` value was set to 100 and the model was trained using the pre-processed dataset and tested on unseen data to make predictions of Bitcoin prices. The trained RF model achieved an RMSE of 0.01973, MSE of 0.00038, MAE of 0.43742, R^2 of 0.45762, TWAP of 0.41585, and VWAP of 0.36904. The evaluation metrics used to assess the trained model, along with the actual and predicted values, are compared in the bar graph 5.1 and line graph 5.2 shown below.

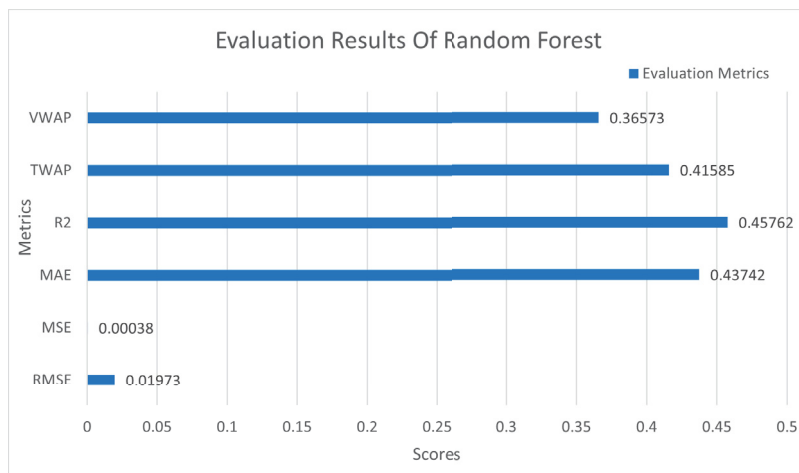


Figure 5.1: Evaluation Results Bar Graph of Random Forest

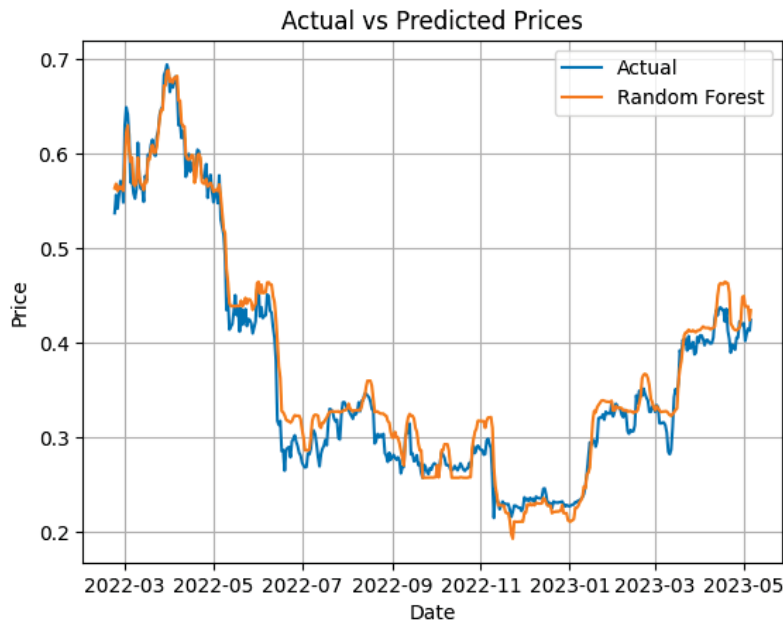


Figure 5.2: Actual Vs Predicted Prices of Random Forest

After the evaluation was done the feature importance was computed in order to understand which feature in the dataset affected the prediction of the model more. The feature importance was measured by performing the method '**permutation feature importance**'. This method is done by randomly permuting the values of a single feature in the test dataset and observing the resulting decrease in the model's performance. The RF's feature importance scores for the features 'RSI', 'SMA', and 'EMA' are 0.00009, 0.01294, and 0.00411, respectively. The feature importance scores of RF are displayed in the form of a bar graph 5.3 below.

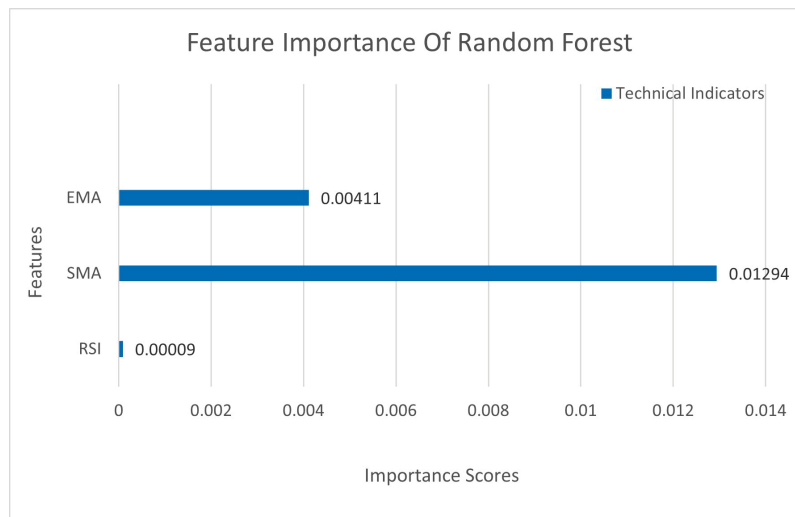


Figure 5.3: Feature Importance Bar Graph of Random Forest

Also, correlation analysis was performed to find out the linear relationship between each feature and the predictions of the RF. The correlation coefficient values

for the features for 'RSI', 'SMA', and 'EMA' with the prediction values of RF are 0.0697, 0.9933, and 0.9929 respectively. The bar graph 5.4 below displays the correlation between each feature and the predictions of the RF.

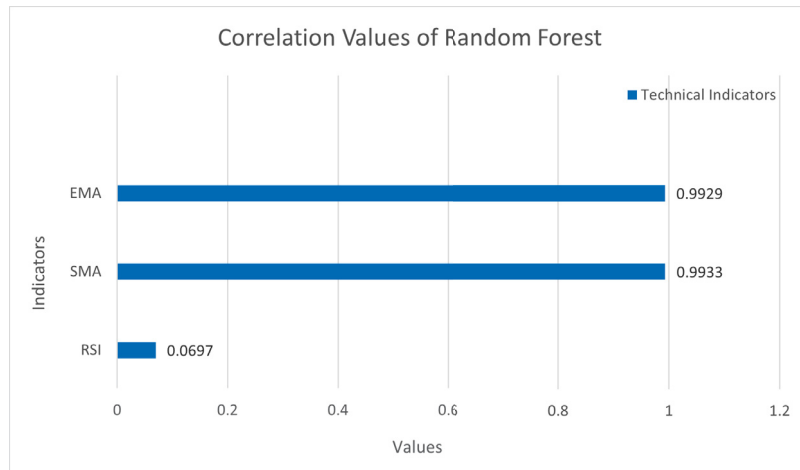


Figure 5.4: Correlation Analysis Bar Graph of Random Forest

Below in figure 5.5 is a snippet of the actual and predicted values of the model.

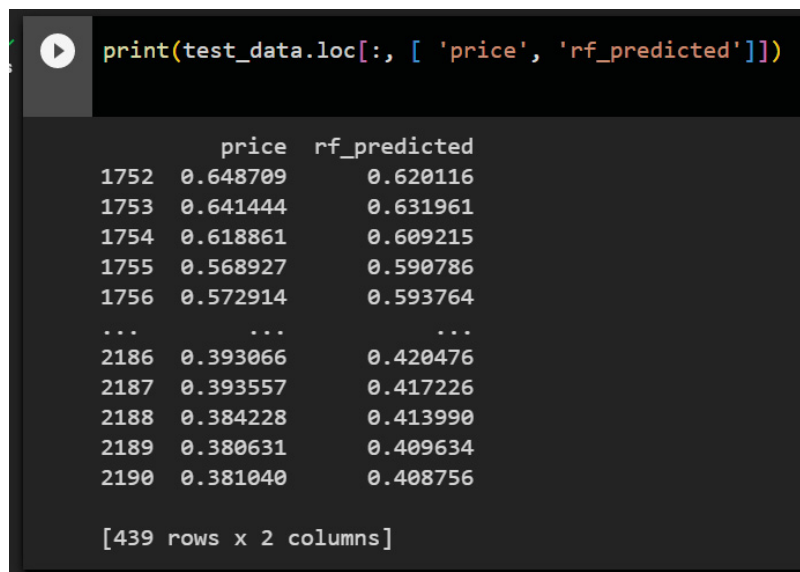


Figure 5.5: Prediction Values of Random Forest Model

5.2.2 Gradient Boosting Regressor

Hyper-parameter tuning was performed on the GB with varying `n_estimators` values and learning rates, it was found that the model performed the best when the `n_estimators` was set to 100 and the learning rate was set to 0.01 with the validation set score of 0.00048. Thereafter, the model was trained using the pre-processed dataset and tested on unseen data to make predictions of Bitcoin prices. The trained

GB achieved an RMSE of 0.01515, MSE of 0.00022, MAE of 0.45006, R^2 of 0.50735, TWAP of 0.41510, and VWAP of 0.36721. The evaluation metrics used to assess the trained model, along with the actual and predicted values, are compared in the bar graph 5.6 and line graph 5.7 shown below.

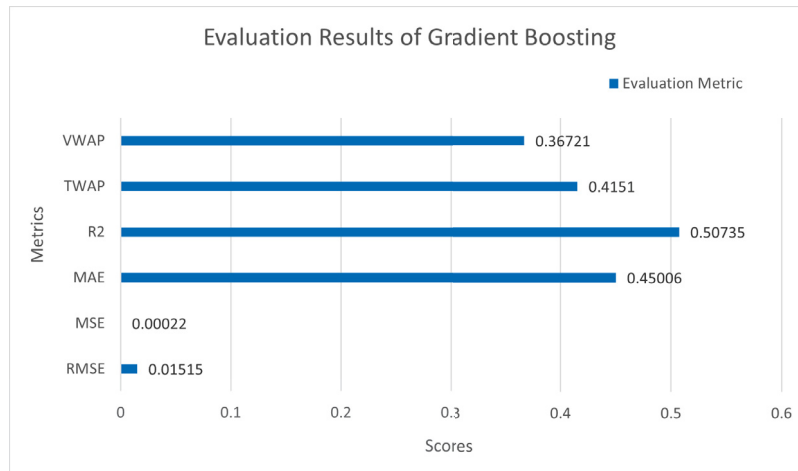


Figure 5.6: Evaluation Results Bar Graph of Gradient Boosting

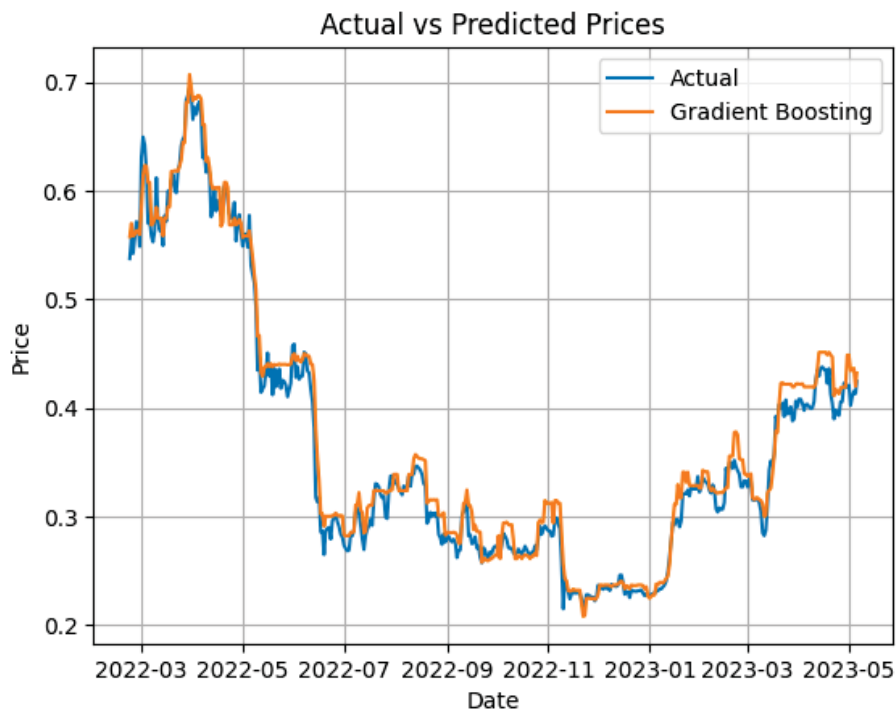


Figure 5.7: Actual Vs Predicted Prices of Gradient Boosting

After the evaluation was done the feature importance was computed in order to understand which feature in the dataset affected the prediction of the model more. The feature importance was measured by performing the method '**permutation feature importance**'. This method is done by randomly permuting the values of a single

feature in the test dataset and observing the resulting decrease in the model's performance. The GB's feature importance scores for the features 'RSI', 'SMA', and 'EMA' are 0.00015, 0.02495, and 0.00023, respectively. The feature importance scores of the GB are displayed in the form of a bar graph 5.8 below.

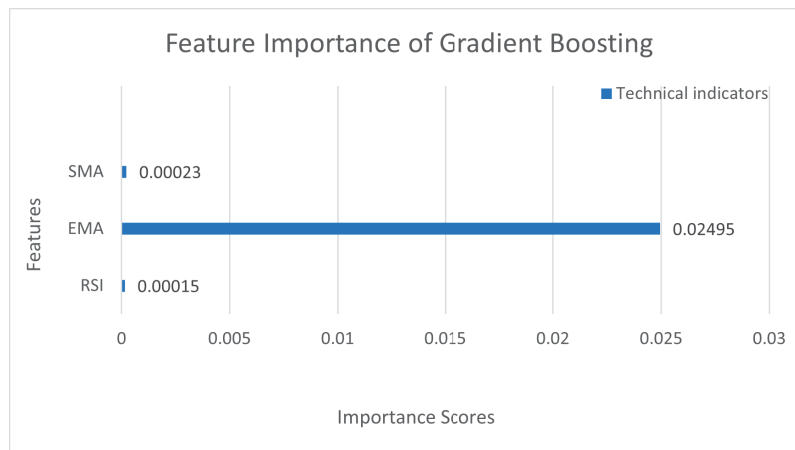


Figure 5.8: Feature Importance Bar Graph of Gradient Boosting

Also, correlation analysis was performed to find out the linear relationship between each feature and the predictions of the GB. The correlation coefficient values for the features for 'RSI', 'SMA', and 'EMA' with the prediction values of GB are 0.0941, 0.9959, and 0.9954 respectively. The bar graph 5.9 below displays the correlation between each feature and the predictions of the GB.

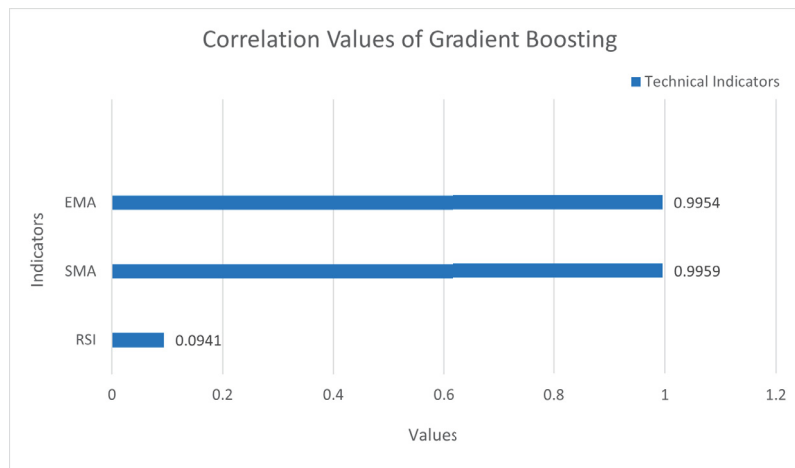


Figure 5.9: Correlation analysis Bar Graph of Gradient Boosting

Below in figure 5.10 is a snippet of the actual and predicted values of the model.



Figure 5.10: Prediction Values of Gradient Boosting Model

5.2.3 Sequential LSTM

Hyper-parameter tuning was performed on LSTM with varying `lstm_units`, `epochs`, and `batch_size` values, it was found that the model performed the best when the `lstm_units` was set to 50, `batch_size` to 32, and `epochs` to 50 with the validation set score of 0.00050. Thereafter, the Sequential LSTM was trained using the pre-processed dataset and tested on unseen data to make predictions of Bitcoin prices. The trained LSTM model achieved an RMSE of 0.01083, MSE of 0.00011, MAE of 0.80635, R^2 of 0.80618, TWAP of 0.40507, and VWAP of 0.35660. The evaluation metrics used to assess the trained model, along with the actual and predicted values, are compared in the bar graph 5.11 and line graph 5.12 shown below.

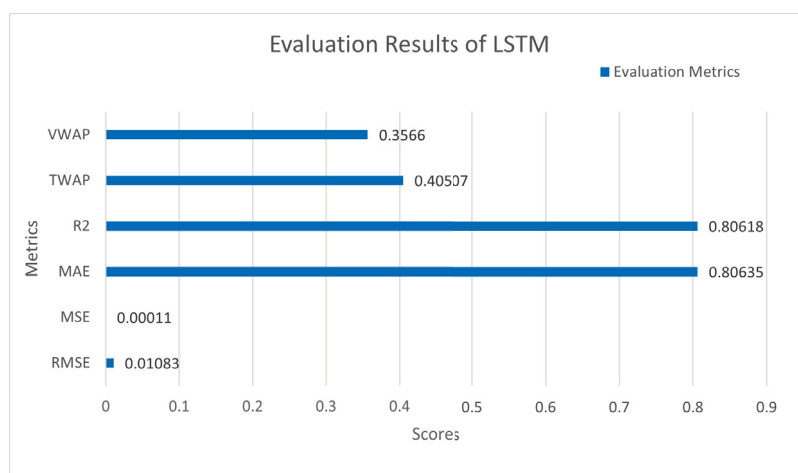


Figure 5.11: Evaluation Results Bar Graph of LSTM

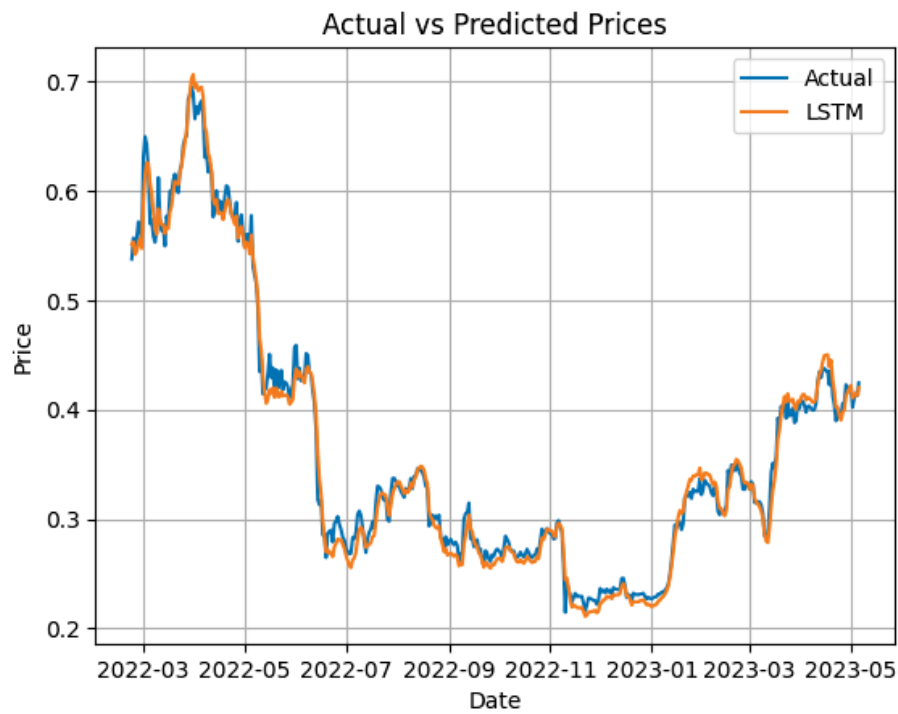


Figure 5.12: Actual Vs Predicted Prices of LSTM

After the evaluation was done the feature importance was computed in order to understand which feature in the dataset affected the prediction of the model more. The feature importance is measured by performing the method '**permutation feature importance**'. This method is done by randomly permuting the values of a single feature in the test dataset and observing the resulting decrease in the model's performance. The LSTM model's feature importance scores for the features 'RSI', 'SMA', and 'EMA' are 0.00019, 0.00665, and 0.00824, respectively. The feature importance scores of the LSTM model are displayed in the form of a bar graph 5.13 below.

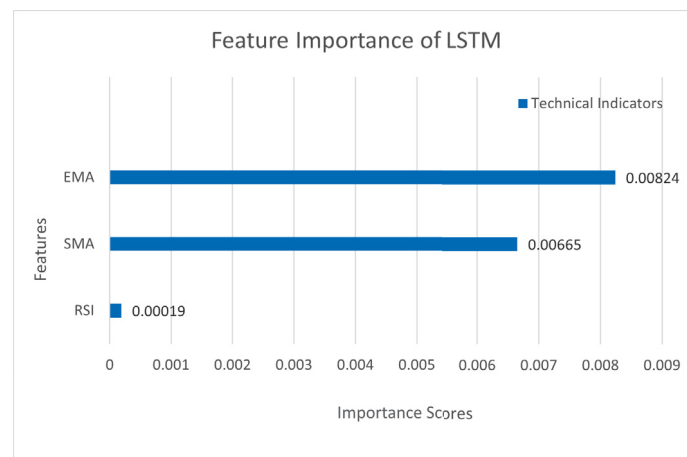


Figure 5.13: Feature Importance Bar Graph of LSTM

Also, correlation analysis was performed to find out the linear relationship between each feature and the predictions of the LSTM. The bar graph 5.14 below displays the correlation between each feature and the predictions of the Sequential LSTM model. The correlation coefficient values for the features for 'RSI', 'SMA', and 'EMA' with the prediction values of the LSTM model are 0.1231, 0.9966, and 0.9963 respectively.

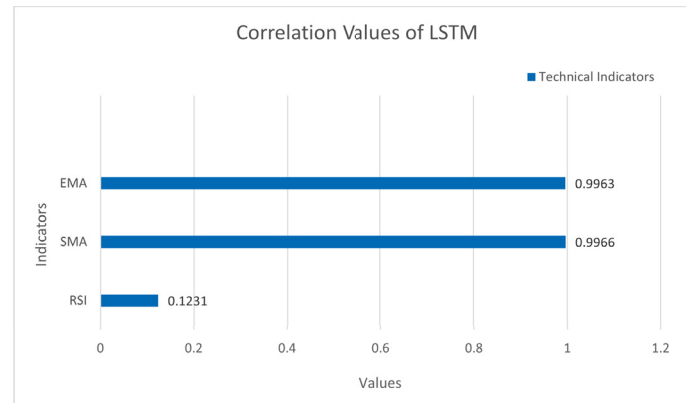


Figure 5.14: Correlation Analysis Bar Graph of LSTM

Below in figure 5.15 is a snippet of the actual and predicted values of the model.

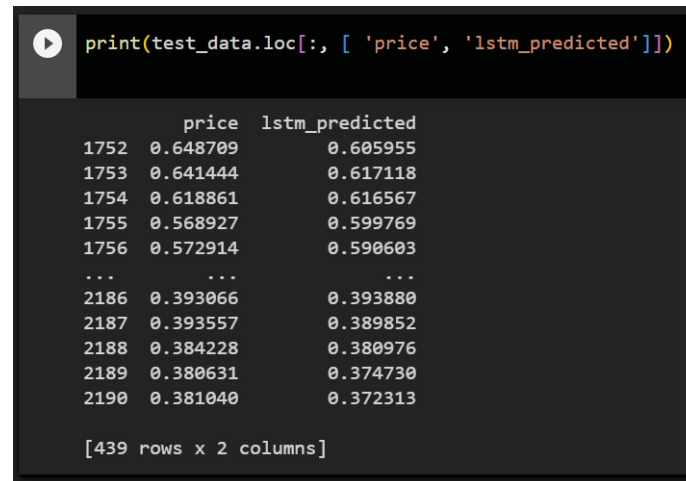


Figure 5.15: Prediction Values of LSTM Model

5.2.4 Sequential GRU

Hyper-parameter tuning was performed on GRU with varying `gru_units`, `epochs` and `batch_size` values, it was found that the model performed the best when the `gru_units` was set to 50, `batch_size` to 32 and `epochs` to 50 with the validation set score of 0.00055. Thereafter, the Sequential GRU was trained using the pre-processed

dataset and tested on unseen data to make predictions of Bitcoin prices. The trained GRU model achieved an RMSE of 0.01271, MSE of 0.00016, MAE of 0.80670, R^2 of 0.80627, TWAP of 0.41330, and VWAP of 0.36573. The evaluation metrics used to assess the trained model, along with the actual and predicted values, are compared in the bar graph 5.16 and line graph 5.17 shown below.

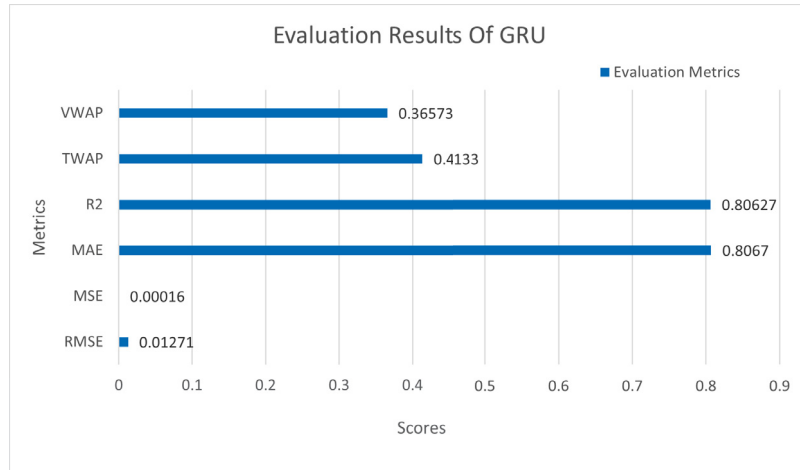


Figure 5.16: Evaluation Results Bar Graph of GRU

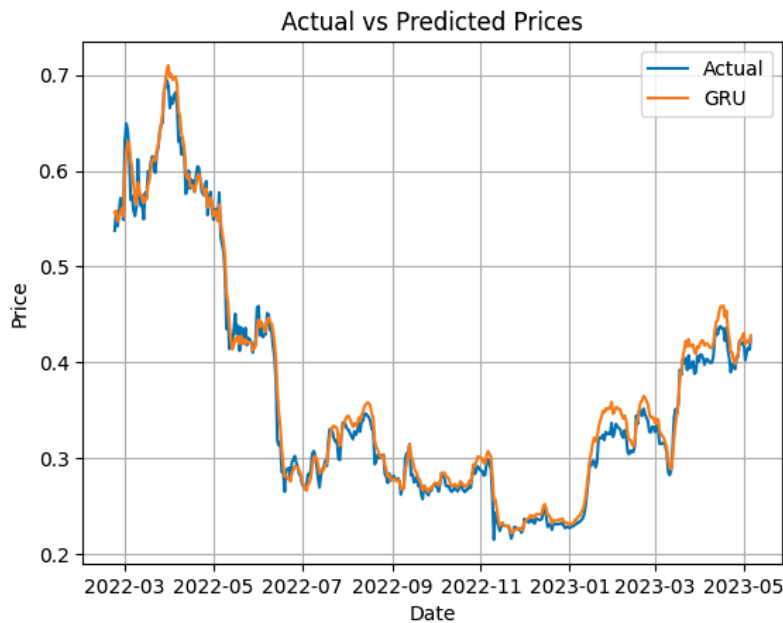


Figure 5.17: Actual Vs Predicted Prices of GRU

After the evaluation was done the feature importance was computed in order to understand which feature in the dataset affected the prediction of the model more. The feature importance is measured by performing the method '**permutation feature importance**'. This method is done by randomly permuting the values of a single

feature in the test dataset and observing the resulting decrease in the model's performance. The GRU model's feature importance scores for the features 'RSI', 'SMA', and 'EMA' are 0.00019, 0.00528, and 0.00988, respectively. The feature importance scores of the GRU model are displayed in the form of a bar graph 5.18 below.

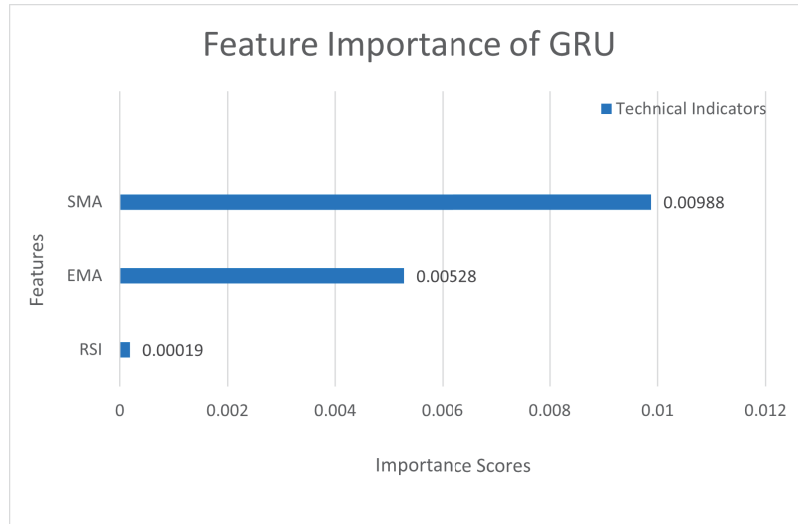


Figure 5.18: Feature Importance Bar Graph of GRU

Also, correlation analysis was performed to find out the linear relationship between each feature and the predictions of the GRU. The correlation coefficient values for the features of 'RSI', 'SMA', and 'EMA' with the prediction values of the GRU model are 0.1284, 0.9961, and 0.9959 respectively. The bar graph 5.19 below displays the correlation between each feature and the predictions of the Sequential GRU model.

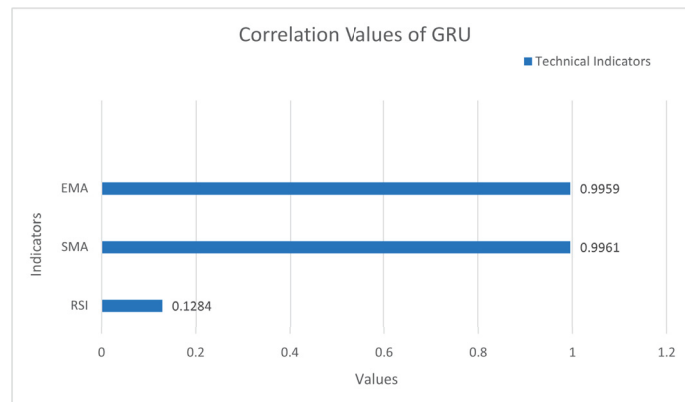
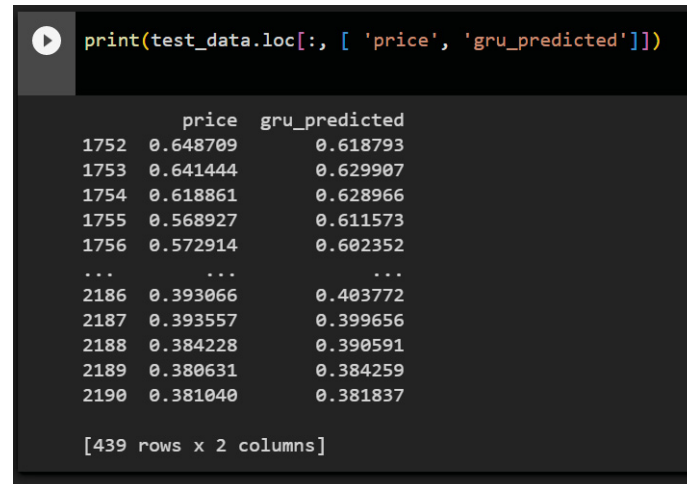


Figure 5.19: Correlation Analysis Bar Graph of GRU

Below in figure 5.20 is a snippet of the actual and predicted values of the model.



```
print(test_data.loc[:, ['price', 'gru_predicted']])
```

	price	gru_predicted
1752	0.648709	0.618793
1753	0.641444	0.629907
1754	0.618861	0.628966
1755	0.568927	0.611573
1756	0.572914	0.602352
...
2186	0.393066	0.403772
2187	0.393557	0.399656
2188	0.384228	0.390591
2189	0.380631	0.384259
2190	0.381040	0.381837

[439 rows x 2 columns]

Figure 5.20: Prediction Values of GRU Model

5.3 Analysis

The results gathered from the aforementioned sections are the following:

Based on the Evaluation Metric Scores:

- Based on the RMSE scores, LSTM was the best-performing model.
- Based on the MSE scores, LSTM was the best-performing model.
- Based on the MAE scores, RF was the best-performing model.
- Based on the R^2 scores, GRU was the best-performing model.
- Based on the TWAP scores, LSTM was the best-performing model.
- Based on the VWAP scores, LSTM was the best-performing model.

Based on the permutation feature importance scores:

- The technical indicator RSI affected the prediction power of both GRU and LSTM the most.
- The technical indicator EMA affected the prediction power of GB the most.
- The technical indicator SMA affected the prediction power of GRU the most

Based on the correlation analysis scores:

- The technical indicator RSI affected the prediction power of GRU the most.
- The technical indicator EMA affected the prediction power of LSTM the most.
- The technical indicator SMA affected the prediction power of LSTM the most.

A SLR was conducted to identify the ML algorithms suitable for the prediction of Bitcoin prices. Then an experiment was conducted using the algorithms identified from the performed SLR. For the experiment historic price data of Bitcoin in USD was collected. Technical indicators were calculated and a dataset was created using these indicators as input features and the prices as the target variable.

The dataset was preprocessed and was used along with the hyper-parameter values found via hyper-parameter tuning to train each ML model. The trained ML models were tested using unseen testing data. The predictions of each model were evaluated using the chosen metrics. Permutation feature importance and correlation analysis were performed on the tested models to identify the importance of each feature in predictions.

The intention behind conducting this thesis was to find out an ideal machine learning algorithm for the prediction of Bitcoin prices. In this section, the research questions are answered.

6.1 Research Question 1

Which prediction models (ML algorithms) are better suited to predict the prices of Bitcoin based on technical indicators calculated using historical price data?

This question is answered by using the research methodology, SLR done in section 5.1. To conduct the SLR, relevant literature was gathered following the inclusion and exclusion principles mentioned in section 5.1. The databases Google Scholar, IEE-Explore, Science Direct, and arXiv were scoured to procure the required literature. After the analysis of the gathered literature, many ML algorithms were identified. Out of which, LSTM and GRU were found to have better predictive accuracy when compared to other machine learning algorithms [4, 29, 30, 42, 46]. It was also found that ensemble methods like RF and GB regressors were highly effective in finding out the future trends of cryptocurrencies [11, 25, 34, 47]. RF and GB algorithms combine individual models to make predictions. LSTM and GRU are DL neural networks that make use of a memory cell component to make predictions. Therefore, the experiment was conducted using the ML algorithms: RF, GB, LSTM, and GRU. The findings of the SLR are reported in table 5.1.

6.2 Research Question 2

Which among the selected ML algorithms performs the best in predicting the price of Bitcoin when using technical indicators (SMA, EMA, and RSI) as input, and which among these indicators affects the prediction performance of each ML algorithm the most?

An experiment was conducted using the algorithms identified in RQ1. Prediction Models were built using the chosen algorithms. Firstly historic price data of Bitcoin in USD was collected using the CoinGecko API. The volume, price data, and their corresponding timestamps were requested using the API endpoint. Three technical indicators RSI, EMA, and SMA were calculated using the requested prices data. A dataset containing 6 columns: Time, Price, Volume, RSI, EMA, and SMA was created. Then data preprocess takes place where all the missing and NaN values are replaced with 0, and all the data is scaled to a range of 0 to 1. This process is done to ensure that there are no biased or inaccurate results and to ensure that all the features have similar magnitude.

After the dataset is preprocessed it is divided into training, validation, and testing tests. 70% of the data is allocated for the training set, 10% for validation, and 20% for the testing set. Hyper-parameter tuning was performed to find out the best parameter configuration for getting efficient results. The value of the hyper-parameter `n_estimators` when increased from 50 to 100 was found to improve the performance of RF and GB models. On further increasing the values to above 150, it was found that the performance started to decrease, hence the `n_estimators` was set to 100 in both the regressors. In the case of LSTM and GRU models optimal prediction performance was found when the `batch_size` was set to 32, epochs to 50 and the `lstm_units` and `gru_units` set to 50. When these hyper-parameter values were increased beyond the stated values the performance was found to lacking. Therefore, the hyper-parameters chosen for LSTM and GRU models were: `batch_size` of 32, epochs of 50, and `lstm_units` and `gru_units` of 50.

Thereafter, each model is trained using the training dataset. After the training of the models, each trained model is tested using the unseen dataset. The predictions of each model are evaluated using the evaluation metrics: RMSE, MSE, MAE, R^2 , TWAP, and VWAP.

The predictions made by each model are scaled predictions due to the application of the `MinMaxScaler()` function in the data preprocessing step. The actual format of the prices can be achieved with the help of the `inverse_transform()` function which is provided by the `MinMaxScaler()` itself. This procedure was not done in this thesis as the main aim was to find out how well each model was predicting the prices rather than the prediction of exact prices itself. We achieve the same by comparing the scaled predictions with the scaled actual prices.

To find which feature affected the prediction of each model more permutation feature importance and correlation analysis between each feature and the prediction

values of all the models is performed. The below table 6.1 shows the evaluation results of all the models.

Model	RMSE	MSE	MAE	R^2	TWAP	VWAP
RF	0.01973	0.00038	0.43742	0.45762	0.41585	0.36904
GB	0.01515	0.00022	0.45006	0.50735	0.41510	0.36721
LSTM	0.01083	0.00011	0.80635	0.80618	0.40507	0.35660
GRU	0.01271	0.00016	0.80670	0.80627	0.41330	0.36573

Table 6.1: Evaluation Metric Values

From the results, it can be determined that LSTM and GRU models have better accuracy in predicting Bitcoin prices when compared to the RF and GB models. Since, they have the lowest RMSE (0.01083, 0.01271) and (0.00011, 0.00016) in particular LSTM having the lowest RMSE value performs the best among the chosen models. Even though the RF and GB models have slightly higher RMSE (0.01973, 0.01515) and MSE(0.00038, 0.00022), they still perform relatively well in terms of accuracy. However, since RMSE measures the average deviation of the predictions from the actual values it can be sensitive to outliers. Other metrics have also been considered to evaluate the models for more informed findings. From the results reported it could also be argued that RF and GB having lower MAE values (0.43742, 0.45006) as compared to LSTM (0.80635) and GRU (0.80670), suggests that RF performs better when compared to other models in terms of MAE scores and has a lower absolute difference between the predicted price and the actual price on an average when compared with other models.

Another possible explanation based on the R^2 metric may be that LSTM (0.80618) and GRU (0.80627) are better able to explain the variance in the target variable than the RF (0.45762) and GB (0.50735) models. The TWAP and VWAP scores indicate that in long-term predictions the LSTM (0.40507, 0.35660) and GRU (0.41330, 0.36573) models perform slightly better than the RF (0.41585, 0.36904) and GB (0.41510, 0.36721) models since their TWAP and VWAP scores are closer to the actual TWAP and VWAP (0.39953, 0.34847) values. Therefore, based on the evaluation metrics it can be said that the **LSTM** model having better RMSE, MSE, and R^2 scores performs the **best** in making long-term price predictions of Bitcoin using technical indicators.

After conducting permutation feature importance, the combined feature importance scores of each model are shown in table 6.2 below:

Model	RSI	EMA	SMA
RF	0.00009	0.01294	0.00411
GB	0.00015	0.02495	0.00023
LSTM	0.00019	0.00665	0.00824
GRU	0.00019	0.00528	0.00988

Table 6.2: Feature Importance Scores

From the table 6.2, it can be determined that the technical indicator EMA is important for both RF (0.01294) and GB (0.02495), while SMA was the most important feature for LSTM (0.00824) and GRU (0.00988). Overall, it can be concluded that the EMA feature is important for all models, while the importance of SMA is more in LSTM and GRU.

Additionally, after conducting the correlation analysis of each feature with the predictions of each model to find out the linear relationships between the features and the predicted values. It was found that the technical indicator EMA had a very strong correlation in all models, followed by SMA. RSI showed the least correlation. The correlation results are presented in the table 6.3.

Model	RSI	EMA	SMA
RF	0.0697	0.9933	0.9929
GB	0.0941	0.9959	0.9954
LSTM	0.1231	0.9966	0.9963
GRU	0.1284	0.9961	0.9959

Table 6.3: Correlation Analysis Scores

The images that follow show the aggregate outcomes of all of the models' forecasts.

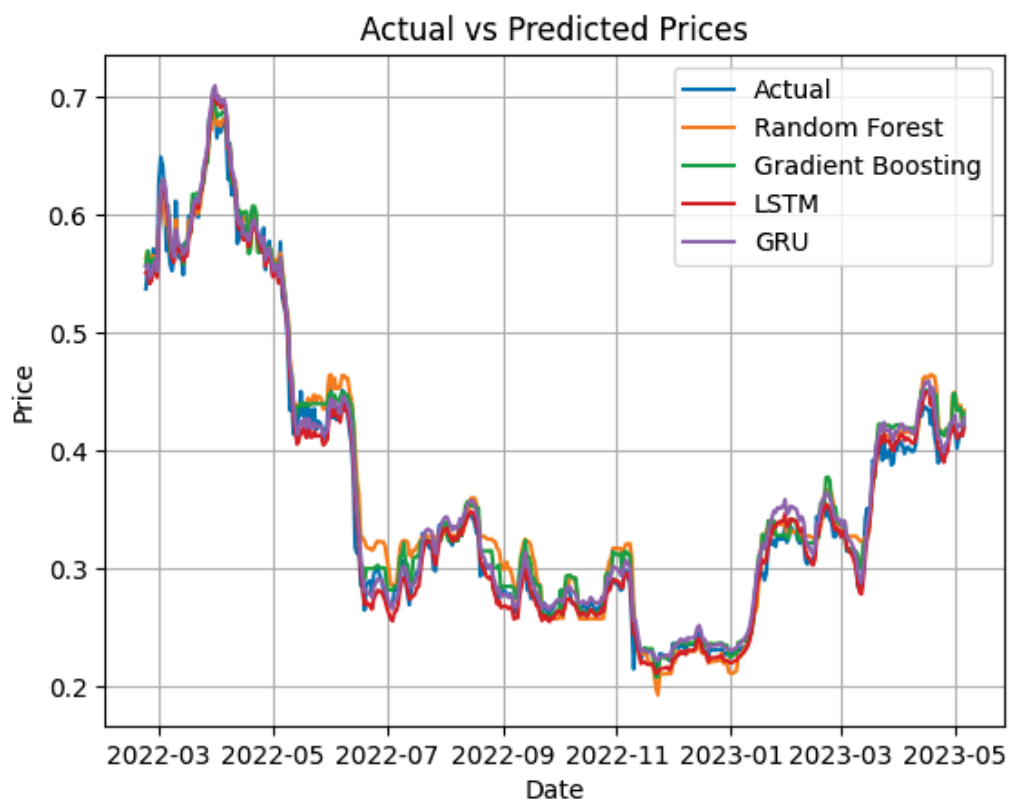


Figure 6.1: Actual Vs Predicted Prices of All models

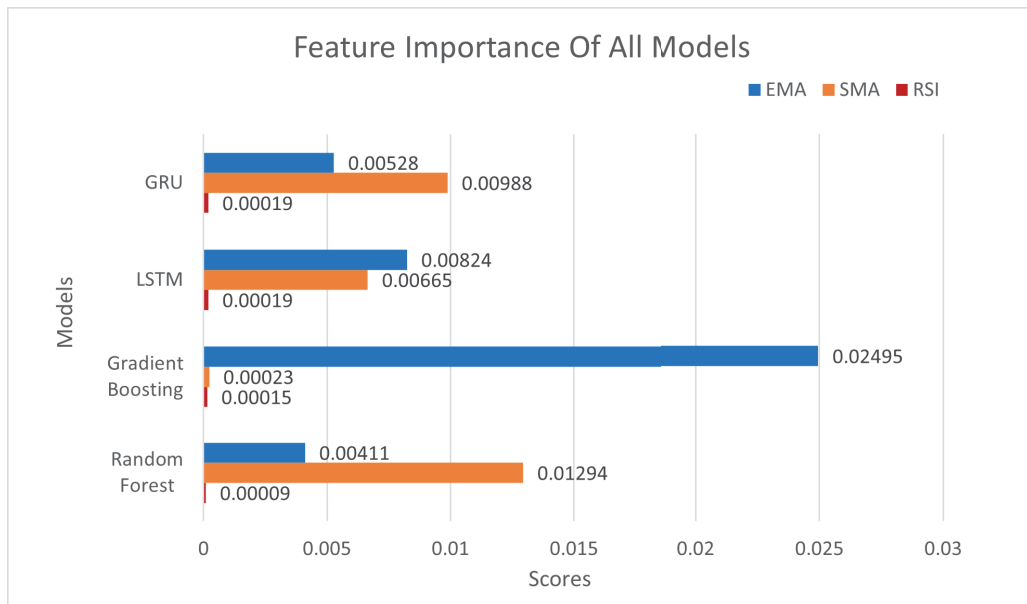


Figure 6.2: Feature Importance Graph of All Models

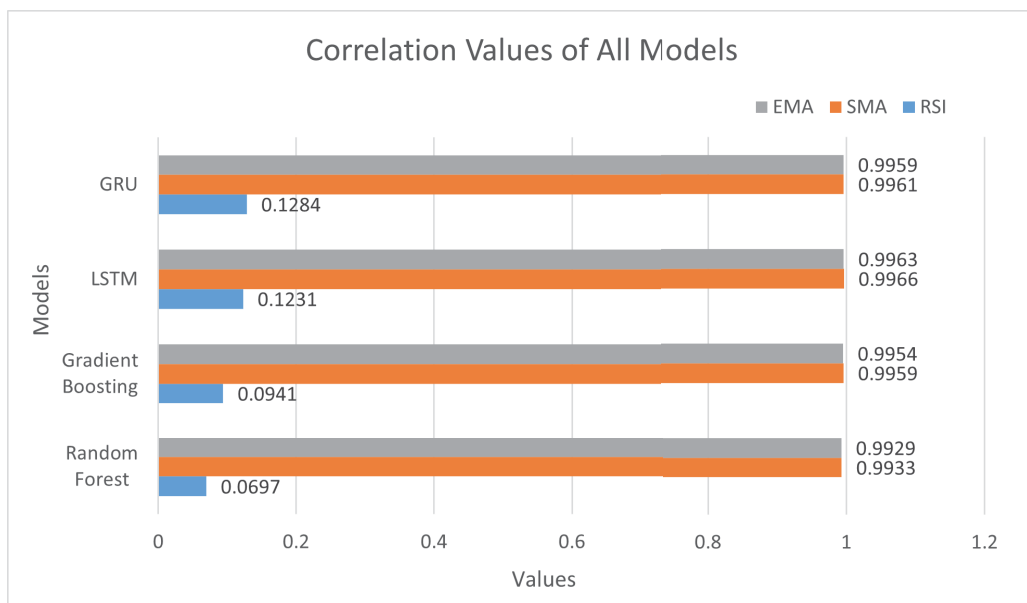


Figure 6.3: Correlation Analysis Graph of All Models

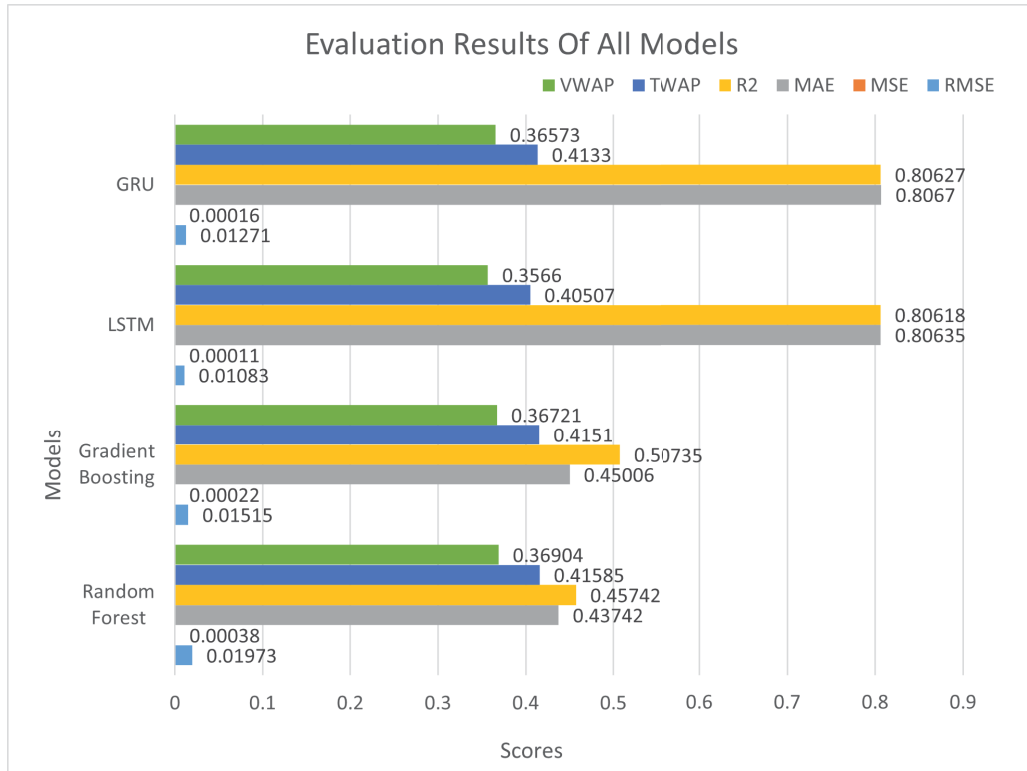


Figure 6.4: Evaluation Results Graph of All Models

By carrying out this experiment, it was discovered that LSTM performed better than the other forecasting models in predicting Bitcoin values. This outcome, however, is restricted to the dataset (CoinGecko) and the technical indicators included in model training. The outcomes can vary if other technical indicators are used.

Chapter 7

Conclusions and Future Work

Public interest in Cryptocurrency as an asset is increasing exponentially in recent years. Financial analysis is being conducted to gain insights for investing in cryptocurrency as the market is quite volatile. Indicators like SMA, EMA, and RSI are normally used in financial analysis. Investors have employed many techniques for gaining more knowledge on future trends of cryptocurrency. One such way adopted is predicting prices using machine learning algorithms. This thesis focuses on conducting a financial analysis by using technical indicators as input features for training machine learning models.

A SLR was conducted for the purpose of choosing the relevant algorithms to conduct the study on. The algorithms chosen after the review were: **RF**, **GB**, **LSTM**, and **GRU**. Optimal hyper-parameters for each model were found by performing hyper-parameter tuning using the validation dataset. The models were then trained on the technical indicators calculated using historic price data of Bitcoin in USD. The trained models were tested and evaluated using various evaluation metrics to gain insights into the performance of each ML algorithm. After comparing the metric scores of each model, **LSTM** with **RMSE** score (0.01083), **MSE** score (0.00011), **R²** score (0.80618), **TWAP** score (0.40507), and **VWAP** score (0.35660) was found to be the **best-performing model** in predicting the Bitcoin prices when compared to the other models. By performing permutation feature importance and correlation analysis it was found that the **moving averages EMA and SMA** had a **greater impact** on the performance of the prediction models as compared to RSI. The results obtained from conducting this thesis are specific to the dataset and the technical indicators used to train the models.

In the future, we would like to modify the dataset by taking a larger dataset, considering other technical indicators, testing on a different dataset, and considering economic and social factors. We would also like to consider other cryptocurrencies to conduct this experiment on. Building prediction models using other algorithms and other feature importance techniques to identify the significance of each input feature on the prediction capabilities of the machine learning models.

References

- [1] “Bitcoin (BTC) Price, Charts, and News | Coinbase: bitcoin price, bitcoin coinbase, bitcoin.” [Online]. Available: <https://www.coinbase.com/price/bitcoin>
- [2] “CoinGecko.” [Online]. Available: <https://support.coingecko.com/hc/en-us>
- [3] “TWAP vs. VWAP Price Algorithms | Chainlink.” [Online]. Available: <https://chain.link/education-hub/twap-vs-vwap>
- [4] “AN APPLIED STUDY OF RNN MODELS FOR PREDICTING CRYPTOCURRENCY PRICES,” *Issues In Information Systems*, 2020. [Online]. Available: https://iacis.org/iis/2020/2_iis_2020_135-143.pdf
- [5] Z. Ai and Z. Yao, “The Investment Value and the Current Regulation of Cryptocurrencies Market Under the Confusion,” Jan. 2021. [Online]. Available: https://www.researchgate.net/publication/357619082_The_Investment_Value_and_the_Current_Regulation_of_Cryptocurrencies_Market_Under_the_Confusion
- [6] M. Ali and S. Shatabda, “A data selection methodology to train linear regression model to predict bitcoin price,” in *2020 2nd International Conference on Advanced Information and Communication Technology (ICAICT)*, 2020, pp. 330–335. [Online]. Available: https://www.researchgate.net/publication/348987864_A_Data_Selection_Methodology_to_Train_Linear_Regression_Model_to_Predict_Bitcoin_Price
- [7] A. F. Bariviera, M. J. Basgall, W. Hasperué, and M. Naiouf, “Some stylized facts of the bitcoin market,” *Physica A: Statistical Mechanics and its Applications*, vol. 484, pp. 82–90, 2017. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0378437117304697>
- [8] G. Biau and E. Scornet, “A random forest guided tour,” *TEST*, vol. 25, no. 2, pp. 197–227, Jun. 2016. [Online]. Available: <https://doi.org/10.1007/s11749-016-0481-7>
- [9] Z. Chen, C. Li, and W. Sun, “Bitcoin price prediction using machine learning: An approach to sample dimension engineering,” *Journal of Computational and Applied Mathematics*, vol. 365, p. 112395, Feb. 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S037704271930398X>
- [10] F. Chollet *et al.* (2015) Keras. [Online]. Available: <https://github.com/fchollet/keras>
- [11] R. Chowdhury, M. A. Rahman, M. S. Rahman, and M. R. C. Mahdy, “An approach to predict and forecast the price of constituents and index

- of cryptocurrency using machine learning,” *Physica A: Statistical Mechanics and its Applications*, vol. 551, p. 124569, Aug. 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0378437120302703>
- [12] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,” Dec. 2014, arXiv:1412.3555 [cs]. [Online]. Available: <http://arxiv.org/abs/1412.3555>
- [13] L. Cocco, R. Tonelli, and M. Marchesi, “Predictions of bitcoin prices through machine learning based frameworks,” *PeerJ Computer Science*, vol. 7, p. e413, Mar. 2021, publisher: PeerJ Inc. [Online]. Available: <https://peerj.com/articles/cs-413>
- [14] CoinGecko, “CoinGecko privacy policy.” [Online]. Available: <https://www.coingecko.com/en/privacy>
- [15] L. Deng and D. Yu, “Deep Learning: Methods and Applications,” *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, Jun. 2014, publisher: Now Publishers, Inc. [Online]. Available: <https://www.nowpublishers.com/article/Details/SIG-039>
- [16] V. Derbentsev, N. Datsenko, V. Babenko, O. Pushko, and O. Pursky, “Forecasting cryptocurrency prices using ensembles-based machine learning approach,” in *2020 IEEE International Conference on Problems of Infocommunications. Science and Technology (PIC ST)*, 2020, pp. 707–712. [Online]. Available: <https://ieeexplore.ieee.org/document/9468090>
- [17] A. Dutta, S. Kumar, and M. Basu, “A Gated Recurrent Unit Approach to Bitcoin Price Prediction,” *Journal of Risk and Financial Management*, vol. 13, no. 2, p. 23, Feb. 2020, number: 2 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: <https://www.mdpi.com/1911-8074/13/2/23>
- [18] S. Fafalios, P. Charonyktakis, and I. Tsamardinos, “Gradient Boosting Trees,” 2020. [Online]. Available: <https://www.semanticscholar.org/paper/Gradient-Boosting-Trees-Fafalios-Charonyktakis/0fb3ea765ea080a5afa03a3d8ef0fcd8e52c2544>
- [19] N. Gradojevic, D. Kukolj, R. Adcock, and V. Djakovic, “Forecasting Bitcoin with technical analysis: A not-so-random forest?” *International Journal of Forecasting*, vol. 39, no. 1, pp. 1–17, Jan. 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0169207021001230>
- [20] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, “Lstm: A search space odyssey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/7508408>
- [21] B. I. Hameed, “Blockchain and Cryptocurrencies Technology: a survey,” *JOIV : International Journal on Informatics Visualization*, vol. 3, no. 4, Nov. 2019. [Online]. Available: <http://joiv.org/index.php/joiv/article/view/293>

- [22] Y. Hari and L. P. Dewi, “Forecasting System Approach for Stock Trading with Relative Strength Index and Moving Average Indicator,” *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 10, no. 2-3, pp. 25–29, May 2018, number: 2-3. [Online]. Available: <https://jtec.utem.edu.my/jtec/article/view/4188>
- [23] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, “Array programming with NumPy,” *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: <https://doi.org/10.1038/s41586-020-2649-2>
- [24] S. Hiregoudar, “Ways to Evaluate Regression Models,” Mar. 2022. [Online]. Available: <https://towardsdatascience.com/ways-to-evaluate-regression-models-77a3ff45ba70>
- [25] J.-Z. Huang, W. Huang, and J. Ni, “Predicting bitcoin returns using high-dimensional technical indicators,” *The Journal of Finance and Data Science*, vol. 5, no. 3, pp. 140–155, Sep. 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2405918818300928>
- [26] J. D. Hunter, “Matplotlib: A 2d graphics environment,” *Computing in Science Engineering*, vol. 9, no. 3, pp. 90–95, 2007. [Online]. Available: <https://ieeexplore.ieee.org/document/4160265>
- [27] M. Iqbal, M. Iqbal, F. Jaskani, K. Iqbal, and A. Hassan, “Time-Series Prediction of Cryptocurrency Market using Machine Learning Techniques,” *EAI Endorsed Transactions on Creative Technologies*, vol. 8, no. 28, p. 170286, Aug. 2021. [Online]. Available: <http://eudl.eu/doi/10.4108/eai.7-7-2021.170286>
- [28] M. A. Istiaque Sunny, M. M. S. Maswood, and A. G. Alharbi, “Deep learning-based stock price prediction using lstm and bi-directional lstm model,” in *2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES)*, 2020, pp. 87–92. [Online]. Available: <https://ieeexplore.ieee.org/document/9257950>
- [29] J. Kim, S. Kim, H. Wimmer, and H. Liu, “A cryptocurrency prediction model using lstm and gru algorithms,” in *2021 IEEE/ACIS 6th International Conference on Big Data, Cloud Computing, and Data Science (BCD)*, 2021, pp. 37–44. [Online]. Available: <https://ieeexplore.ieee.org/document/9581397>
- [30] D. Kumar and S. K. Rath, “Predicting the trends of price for ethereum using deep learning techniques,” in *Artificial Intelligence and Evolutionary Computations in Engineering Systems*, S. S. Dash, C. Lakshmi, S. Das, and B. K. Panigrahi, Eds. Singapore: Springer Singapore, 2020, pp. 103–114. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-15-0199-9_9
- [31] Z. Li, D. He, F. Tian, W. Chen, T. Qin, L. Wang, and T.-Y. Liu, “Towards Binary-Valued Gates for Robust LSTM Training,” Jun. 2018, arXiv:1806.02988 [cs, stat]. [Online]. Available: <http://arxiv.org/abs/1806.02988>

- [32] Y. Liu, Z. Li, R. Nekhili, and J. Sultan, "Forecasting cryptocurrency returns with machine learning," *Research in International Business and Finance*, vol. 64, p. 101905, Jan. 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0275531923000314>
- [33] G. Louppe, "Understanding Random Forests: From Theory to Practice," Jun. 2015, arXiv:1407.7502 [stat]. [Online]. Available: <http://arxiv.org/abs/1407.7502>
- [34] H. Lyu, "Cryptocurrency price forecasting: A comparative study of machine learning model in short-term trading," in *2022 Asia Conference on Algorithms, Computing and Machine Learning (CACML)*, 2022, pp. 280–288. [Online]. Available: <https://ieeexplore.ieee.org/document/9852573>
- [35] B. Mahesh, *Machine Learning Algorithms -A Review*, Jan. 2019. [Online]. Available: https://www.researchgate.net/publication/344717762_Machine_Learning_Algorithms_-A_Review
- [36] A. Moroşan, "The relative strength index revisited," *African journal of business management*, vol. 5, p. 5855, Jul. 2011. [Online]. Available: https://www.researchgate.net/publication/267716142_The_relative_strength_index_revisited
- [37] K. Nakagawa and R. Sakemoto, "Cryptocurrency network factors and gold," *Finance Research Letters*, vol. 46, p. 102375, May 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1544612321003779>
- [38] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in Neurorobotics*, vol. 7, 2013. [Online]. Available: <https://www.frontiersin.org/articles/10.3389/fnbot.2013.00021>
- [39] T. pandas development team, "pandas-dev/pandas: Pandas," Feb. 2020. [Online]. Available: <https://doi.org/10.5281/zenodo.3509134>
- [40] Y. K. Pardeshi and P. P. Kale, "Technical analysis indicators in stock market using machine learning: A comparative analysis," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2021, pp. 1–6. [Online]. Available: <https://ieeexplore.ieee.org/document/9580172>
- [41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, no. null, p. 2825–2830, nov 2011. [Online]. Available: <https://dl.acm.org/doi/10.5555/1953048.2078195>
- [42] T. Phaladisailoed and T. Numnonda, "Machine learning models comparison for bitcoin price prediction," in *2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2018, pp. 506–511. [Online]. Available: <https://ieeexplore.ieee.org/document/8534911>

- [43] E. Pintelas, I. Livieris, S. Stavroyiannis, T. Kotsilieris, and P. Pintelas, *Investigating the Problem of Cryptocurrency Price Prediction: A Deep Learning Approach*, 06 2020. [Online]. Available: https://www.researchgate.net/publication/340256591_Investigating_the_Problem_of_Cryptocurrency_Price_Prediction_A_Deep_Learning_Approach
- [44] A. Prasad and A. Seetharaman, "Importance of Machine Learning in Making Investment Decision in Stock Market," *Vikalpa*, vol. 46, no. 4, pp. 209–222, Dec. 2021, publisher: SAGE Publications India. [Online]. Available: <https://doi.org/10.1177/02560909211059992>
- [45] H. N. Rani, "A STUDY ON FORECASTING OF SECURITY PRICES USING RELATIVE STRENGTH INDEX (RSI) SELECTED COMPANIES IN INDIA," vol. 7, no. 2, 2019. [Online]. Available: https://issuu.com/researchpublish/docs/a_study_on_forecasting-8403/s/19850785
- [46] M. Saad, J. Choi, D. Nyang, J. Kim, and A. Mohaisen, "Toward Characterizing Blockchain-Based Cryptocurrencies for Highly Accurate Predictions," *IEEE Systems Journal*, vol. 14, no. 1, pp. 321–332, Mar. 2020, conference Name: IEEE Systems Journal. [Online]. Available: <https://ieeexplore.ieee.org/document/8406859>
- [47] P. Sadorsky, "Predicting Gold and Silver Price Direction Using Tree-Based Classifiers," *Journal of Risk and Financial Management*, vol. 14, no. 5, p. 198, May 2021, number: 5 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: <https://www.mdpi.com/1911-8074/14/5/198>
- [48] R. M. Schmidt, "Recurrent Neural Networks (RNNs): A gentle Introduction and Overview," Nov. 2019, arXiv:1912.05911 [cs, stat]. [Online]. Available: <http://arxiv.org/abs/1912.05911>
- [49] P. L. Seabe, C. R. B. Moutsinga, and E. Pindza, "Forecasting Cryptocurrency Prices Using LSTM, GRU, and Bi-Directional LSTM: A Deep Learning Approach," *Fractal and Fractional*, vol. 7, no. 2, p. 203, Feb. 2023, number: 2 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: <https://www.mdpi.com/2504-3110/7/2/203>
- [50] H. Sebastião and P. Godinho, "Forecasting and trading cryptocurrencies with machine learning under changing market conditions," *Financial Innovation*, vol. 7, no. 1, p. 3, Jan. 2021. [Online]. Available: <https://doi.org/10.1186/s40854-020-00217-x>
- [51] N. Sharma, R. Sharma, and N. Jindal, "Machine Learning and Deep Learning Applications-A Vision," *Global Transitions Proceedings*, vol. 2, no. 1, pp. 24–28, Jun. 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2666285X21000042>
- [52] P. P. Shinde and S. Shah, "A review of machine learning and deep learning applications," in *2018 Fourth International Conference on Computing Communication Control and Automation (ICCCUBEA)*, 2018, pp. 1–6. [Online]. Available: <https://ieeexplore.ieee.org/document/8697857>

- [53] C. Wohlin, “Guidelines for snowballing in systematic literature studies and a replication in software engineering,” in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE '14. New York, NY, USA: Association for Computing Machinery, May 2014, pp. 1–10. [Online]. Available: <https://doi.org/10.1145/2601248.2601268>
- [54] L. Yang and A. Shami, “On hyperparameter optimization of machine learning algorithms: Theory and practice,” *Neurocomputing*, vol. 415, pp. 295–316, Nov. 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0925231220311693>
- [55] Z. Z., P. A. E. A., and H. M. H., “Predicting machine failure using recurrent neural network-gated recurrent unit (RNN-GRU) through time series data,” *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 2, pp. 870–878, Apr. 2021. [Online]. Available: <https://beei.org/index.php/EEI/article/view/2036>
- [56] C. Zhang and P. Woodland, “High Order Recurrent Neural Networks for Acoustic Modelling,” Feb. 2018, arXiv:1802.08314 [cs, eess, stat]. [Online]. Available: <http://arxiv.org/abs/1802.08314>

